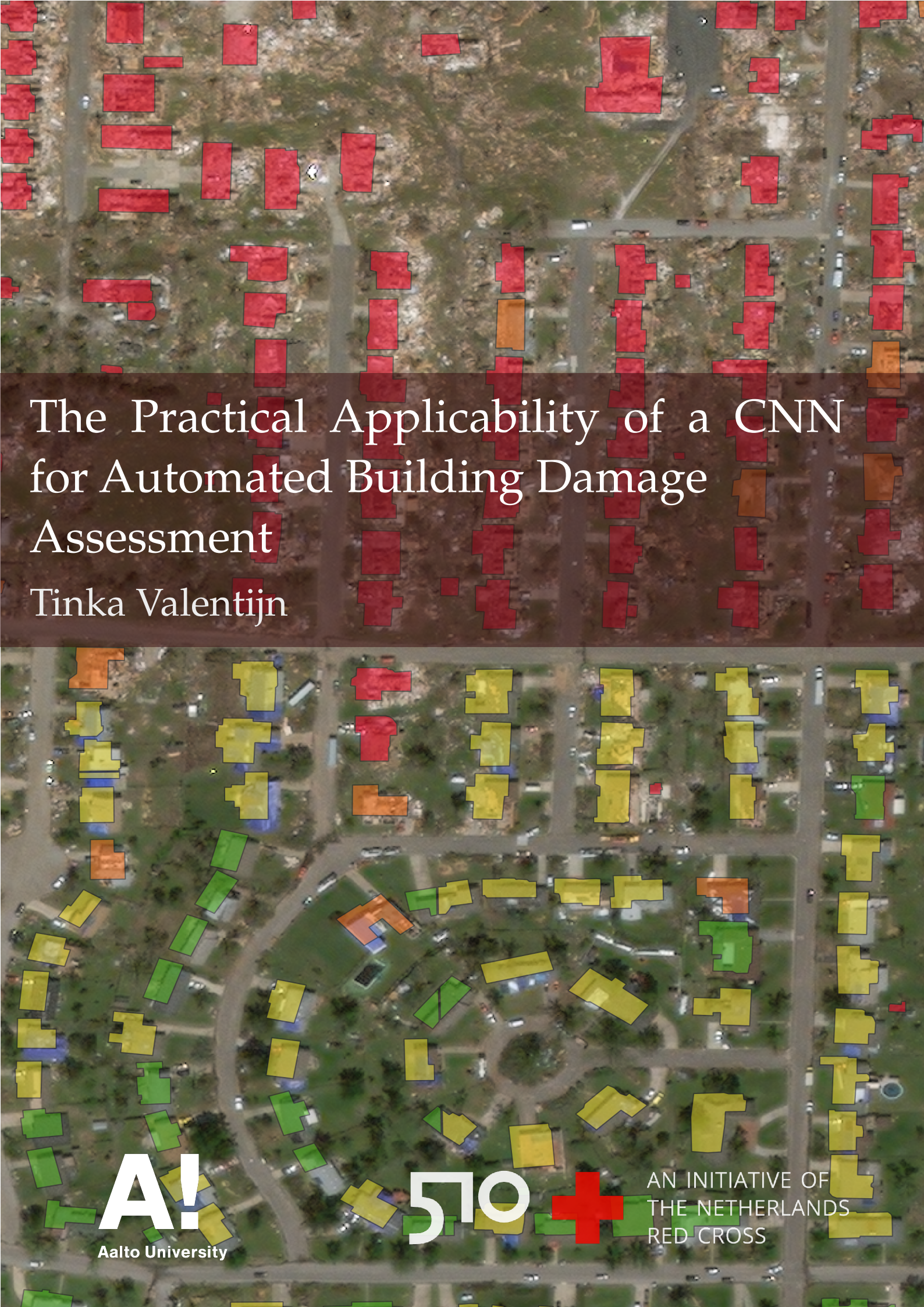


The Practical Applicability of a CNN for Automated Building Damage Assessment

Tinka Valentijn

Aalto University

510

AN INITIATIVE OF THE NETHERLANDS RED CROSS

COVER IMAGE
Satellite imagery after the Joplin tornado overlayed with damage labelled building polygons
(Source satellite imagery: Maxar/DigitalGlobe Open Data Program, used under CC-BY-NC4.0, retrieved from https://www.digitalglobe.com/ecosystem/open-data. Source polygons and labels: xBD dataset, used under CC-BY-NC-SA4.0, retrieved from https://xview2.org/dataset)

CO2 EMISSION RELATED TO EXPERIMENTS
Experiments were conducted using a private infrastructure, which has a carbon efficiency of 0.185 kgCO_2eq/kWh. A cumulative of about 1000 hours of computation was performed on hardware of type Tesla V100 (TDP of 300W). Total emissions are estimated to be 55.5 kgCO_2eq.

Estimations were conducted using the MachineLearning Impact calculator presented in [1].

Aalto University
School of Science
Master's Programme in Computer, Communication and Information Sciences

Tinka Valentijn

The practical applicability of a CNN for automated building damage assessment

Master's Thesis
Espoo, May 11, 2020

Supervisor: Jorma Laaksonen, D.Sc. (Tech.)
Advisors 510: Marc van den Homberg, Dr.
Jacopo Margutti, Dr.
Advisor Aalto: Alexander Jung, Dr.

Aalto University
School of Science
Master's Programme in Computer, Communication and Information Sciences

ABSTRACT OF MASTER'S THESIS

Author: Tinka Valentijn

Title: The practical applicability of a CNN for automated building damage assessment

Date: May 11, 2020 **Pages:** 74

Major: Machine Learning, Artificial Intelligence and Data Science **Code:** SCI3042

Supervisor: Jorma Laaksonen, D.Sc. (Tech.)

Advisors 510: Marc van den Homberg, Dr.
Jacopo Margutti, Dr.

Advisor Aalto: Alexander Jung, Dr.

Every year many people are impacted by the effects of natural hazards. For an effective disaster response, a building damage assessment is of great importance. In current practices, these assessments are done manually. Executing these assessments automatically using satellite imagery holds big potential; it decreases workload while increasing safety, consistency, timeliness, spatial coverage, and possibly accuracy. However, before an automated assessment can be used in real-life situations, it should be well designed to fit user needs. Therefore, this research focuses on the practical applicability of a CNN for automated building damage assessment.

The models' practical applicability is assessed on two requirements: the consistency of performance across disasters; and the performance on a disaster for which no labeled data exists. Firstly, to test the consistency, it is explored how the model performs on a range of 13 disasters, including four different damage types (wind, flooding, tsunami, and volcanic eruptions) across different geographical regions. Testing on this variety of data has never been done before. The results show that performance significantly differs across disasters. Through quantitative analysis it is found that the percentage of buildings belonging to each damage class largely influences the performance, while image and disaster-specific parameters do not show a significant impact.

Secondly, a realistic setting of data availability is simulated, where no labeled damage data of the test disaster is available. A model is trained on several sets of training disasters that do not include the test disaster. The best performance is reached when training solely on disasters that have the same damage type (e.g. wind) as the test disaster. Performance in this set-up differed drastically between the two test disasters experimented with, reaching 93% and 53% of the macro F1 score compared to when the model was trained on the test disaster itself. Thus, the model definitely has potential to learn from previous disasters, but additional research is required to find out what influences the difference in transferability.

Lastly, several modifications to the model were implemented to examine their impact on the performance. The main findings were that the negative impact of data imbalance can be diminished by applying resampling or cost-sensitive learning and that while solely using imagery of the area after the disaster, shows a drop in performance, it could be used in situations where no pre disaster imagery is available.

The experiments in this research show that performance differs significantly across disasters. While for some disasters the model gains good performance, even in the real-life context of not having labeled data of the test disaster, for others the performance is disappointing. A first attempt to understand these differences was made, but further research is needed to affirm the results.

Keywords: damage assessment, building damage, CNN, transfer learning

Language: English

ACKNOWLEDGEMENTS

"Tell me, and I forget. Teach me, and I remember. Involve me, and I learn."
– Xun Kuang, 312-230 BC[1]

The acknowledgements section of a thesis, is the only place to be sentimental. So I hereby use this opportunity to give a big thanks to all the people who have let me involve in this world during the last 25 years.

Who showed me that following your heart is too cheesy to always apply in reality but that there is a big truth in it. Who made me realize that real learning happens by inventing, not solely by copying. Who made me understand that many things are not understood by anyone, which leads to a great opportunity for discovery. This group of people is a diverse group. Of course my family, who always supported my ideas even if I did not believe in them yet. My childhood friends with whom I started to explore the grant possibilities of this world. My ATLAS friends with whom I discovered together that chaos also leads to many opportunities. My Finnish friends who widened my worldview ("no Tinka, India is not solely a country with holy cows"). My friends from my travels who showed me that more is possible than you might think. My friends I made at random places who made me realize that kindness goes a long way.

The thesis was the final part of my official study journey. While life is full of learning, it is an important milestone. Thank you 510 for taking me up as part of the team, showing me the complexity of humanitarian aid, and introducing me to the "dodgy bar". Thank you Jorma for always being there to shine your light on my doubts and questions. Thank you to all of those who supported me mentally, whether it was through daily milk foam breaks, laughing together till our cheeks hurt, online activities during the intelligent lock-down, or listening to my worries regardless of how big or small they were.

I hope you enjoy reading this thesis but most of all, I hope we can continue discovering this beautiful world together. That is it for the sentimental stuff, let us get to the real stuff.

1 Not by Benjamin Franklin as commonly thought, see here

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

AUC	area under the curve ... 31
CNN	convolutional neural network ... 14
DL	deep learning ... 10
EM-DAT	The Emergency Events Database ... 5
ESA	European Space Agency ... 11
GIS	geographic information system ... 14
GSD	ground sample distance ... 10
HCD	human-centered design ... 64
HDX	Humanitarian Data Exchange ... 16
IFRC	International Federation of Red Cross and Red Crescent Societies ... 5
ILSVRC	ImageNet Large Scale Visual Recognition Challenge ... 26
ML	machine learning ... 2
NLP	natural language processing ... 27
PDNA	Post-Disaster Needs Assessment ... 5
ROC	receiver operating characteristic ... 31
SAR	synthetic aperture radar ... 10
SGD	stochastic gradient descent ... 22
UAV	unmanned aerial vehicle ... 6

1 INTRODUCTION

Natural hazards that turned into human disasters have cost many lives. Even more so, people's livelihoods have been affected by them [2]. People lost their loved ones, their businesses, and their homes. We can only speculate about the number of people affected by natural hazards in the future [3, 4, 5, 6], but it is evident that this number is not going to be anywhere close to zero in a foreseeable period.

After a natural hazard, it is essential that the affected population is assisted in the most effective way. It is crucial to find out what this entails, as needs of communities might differ. One of the key aspects to determine what kind of relief aid communities need, is having an accurate picture of the environment, the damaged areas, and the type of damage [7].

In current practices, mapping this damage is done manually, either by field surveys or by people remotely, who manually analyze satellite imagery that has become wider available in recent years. However, these methods of manual mapping require a lot of work, often only cover a limited area, can have a long time delay, cannot be updated easily, and will always be subjective. To reduce these disadvantages and still give an accurate picture of the damage, scholars and practitioners are increasingly interested in automated damage assessment [8]. Using remote sensing techniques, and in recent years machine learning (ML) algorithms, visual features can be detected that indicate damage. This has the potential to result in more accurate, quicker, cheaper, and safer damage assessment than when done manually, thereby increasing the chance of effective relief aid.

After a natural hazard has occurred, there are different types of damage. One important type of damage is damage to buildings. Information on damage to buildings is required for the shelter team to assess how people are affected, whether shelters have to be built, and what materials are needed to reconstruct these buildings. Research has been done on automatic building damage assessment, but to the author's knowledge, no open-source model exists which has been applied for real use cases after a disaster. A handful of models exist that have been tested in a research setting, but none of them have been applied in practice. For a model to be useful, the performance of the model must be of a high enough quality for its purpose. In addition, it should be understood if and how the performance of the model changes across disasters and locations since every disaster is unique. Furthermore, the model should not use labeled damage data of the disaster the model is applied to since this is often not available within the time frame during which the damage assessment is needed.

Understanding the performance across disasters, and solely using data that is available in real situations, are factors that often are not taken into account in current research on automated building damage assessment. This research aims to fill a part of this gap.

1.1 RESEARCH CONTEXT

This research is executed as a master thesis project for the MSc in Machine Learning, Artificial Intelligence and Data Science at Aalto University. This research is done in cooperation with 510. 510 is an initiative of the Netherlands Red Cross with the vision to use data for faster and more effective humanitarian aid [9]. One of

their interest areas is building damage assessment. They have developed a damage assessment tool, which they have applied after hurricane Irma in St. Maarten in 2017. The data on St. Maarten was captured by a drone. Since then, they have been developing this model. The current version has shown to have predictive value on the data of St. Maarten. However, the model has not been used on satellite imagery, other disasters, or without using labeled information of the disaster of interest. These are factors that have to be better understood before the tool can be used in production; and this is where this research comes in. Combining the goals of the study program and 510, this research aims to find a balance between exploring novel machine learning ideas and techniques, as well as matching it to the needs of a humanitarian organization.

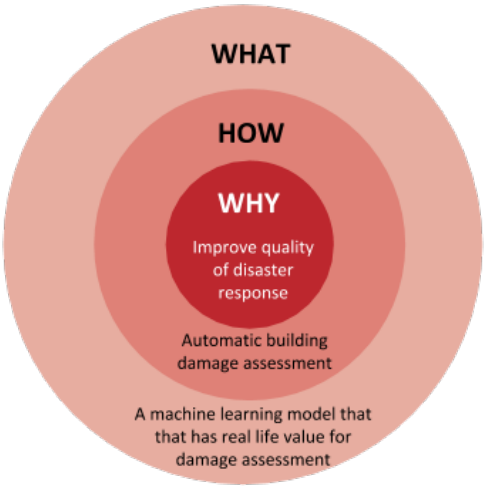


Figure 1.1: The vision of this research.

1.2 RESEARCH GOAL

The goal of this research is to *develop an open-source model for automatic building damage assessment from satellite imagery after disasters caused by sudden-onset natural hazards which is applicable in real-world situations* (Figure 1.1). More specifically, the research focuses on assessing the difference in performance on a range of disasters and experimenting on how the model can be generalizable to perform well on disasters that were not included in the training set. This leads us to the following research questions:

- What is the performance of 510's model on satellite imagery?
- Are there differences in performance between disasters?
- If there are differences, can these be quantified?
- Can good performance be reached when the model is trained on disasters other than the test disaster?
- Do certain modifications to the model improve performance?

To answer the first question, the current model of 510 is adapted to the dataset used in this research. Firstly, this model is used to test the performance on a wide range of disasters. From these results, conclusions of the predictive value of the model can be drawn and potential differences in performance between disasters can be seen. If differences in performance between disasters exist it will be investigated whether or not those differences can be explained through certain parameters, such

as the damage type. Testing on such a wide range of disasters and quantifying their differences, is a proposition that has never been researched before. The second part of this research dives into the question of what the maximum performance is that can be reached when classifying a test disaster without using any labeled data of this disaster. In this research, this implies that the model is solely trained on data from previous disasters. Lastly, it is investigated whether modifications to the model can improve performance, both when training on the test disaster, as well as when the test disaster is excluded from training.

1.3 RESEARCH SCOPE

This research focuses only on disasters caused by natural hazards and where visible building damage occurs. More specifically, five types of disasters are included in this research: hurricanes, tornadoes, floodings, tsunamis, and volcanic eruptions. The data used in this research comes from a dataset named xBD (Section 2.6). The satellite imagery in this dataset is retrieved from the Maxar/DigitalGlobe Open Data Program[1]. This program is activated after natural hazards to make high-resolution imagery available for better relief. This means that the same quality of imagery used in this research is available after a disaster, which is an important requirement for the model to be applicable in the real world.

In practice, automatic damage assessment from satellite imagery consists of two steps. Firstly, the buildings have to be located in the image. Secondly, the damage to those located buildings has to be assessed. Both steps are an active area of research, worldwide as well as within 510 [10, 11]. In order to limit the scope, this research only focuses on the latter, the damage classification. This choice was made since we believe that this is the step where the most improvement can and has to be made. For building localization, more research has already been done on generalization to unseen locations and better open-source tools exist[2]. The available data and 510 environment provide an excellent context to also advance the knowledge on damage classification for practical applicability.

In terms of applicability, this research focuses on the performance of damage classification across disasters in the dataset and without labeled data of the disaster of interest. However, for it to be really applicable in practice, it should also be understood what factors should be taken into account in order for the model to become operational in the humanitarian sector. Since the coverage on the literature for user needs is extremely scarce, the author makes an attempt for a first step towards the understandability of user needs (Section 2.2).

Chapter summary

- Building damage assessment is essential for an effective disaster response
- The goal of this research is to develop an open-source model for automatic building damage assessment from satellite imagery after disasters caused by sudden-onset natural hazards which is applicable in real-world situations
- The specific focus is on performance on a wide range of disasters and transferability of a trained model to a new test disaster

1 https://www.digitalglobe.com/ecosystem/open-data
2 For example *neat-EO* [12]

2 LITERATURE REVIEW

This chapter will explore the context that is needed for automated damage assessment. It will explain concepts related to disaster management, damage assessment, remote sensing, different sources of data, and the current state of the art of automated damage assessment.

2.1 NATURAL HAZARDS AND DISASTERS

The terms hazard, natural hazard, and disaster are often used interchangeably. However, they are not the same and bear important differences that are relevant to the terminology in this research.

The Emergency Events Database (EM-DAT) defines a hazard as *a threatening event, or probability of occurrence of a potentially damaging phenomenon within a given period and area* [13]. This is seen from a human perspective, as many hazards do not have to be threatening to the natural environment itself. A hazard can further be divided into different types of hazards. The International Federation of Red Cross and Red Crescent Societies (IFRC) divides hazards in natural and technological hazards [14]. Natural hazards are defined as *naturally occurring physical phenomena caused either by rapid or slow-onset events* and can be divided to geophysical (earthquakes, landslides, tsunamis, and volcanic activity), hydrological (avalanches and floods), climatological (extreme temperatures, drought, and wildfires), meteorological (cyclones and storms/wave surges) or biological (disease epidemics and insect/animal plagues) [14]. In this research, we only focus on damage caused by natural hazards with rapid-onset and visible damage to buildings.

Natural hazards do not necessarily turn into disasters. Natural hazards combined with the vulnerability and lack of coping capacity of the community gives a certain risk. If the risk is high then the natural hazard can lead to a disaster. IFRC defines a disaster as *a sudden, calamitous event that seriously disrupts the functioning of a community or society and causes human, material, and economic or environmental losses that exceed the community's or society's ability to cope using its resources. Though often caused by nature, disasters can have human origins* [14]. This indicates that the community is not able to solve the consequences themselves, and thus external resources are needed. Natural disasters do not technically exist, but often the term is used to indicate a disaster caused by a natural hazard.

2.2 DAMAGE ASSESSMENT

This research focuses on the damage assessment of buildings. Damage assessment is key for good disaster management after disasters with visual damage to the infrastructure [7]. Disaster management is often divided into four phases: mitigation, preparedness, response, and recovery, as illustrated in Figure 2.1. Damage assessment is mainly relevant in the response and recovery phases. Many organizations and governments assess the damage, but all of them have their own scope, purpose, and methods [5, 8]. The scope of these assessments ranges from very quick estimations of the damaged areas to elaborate reports, such as the Post-Disaster Needs Assessment (PDNA) [16].

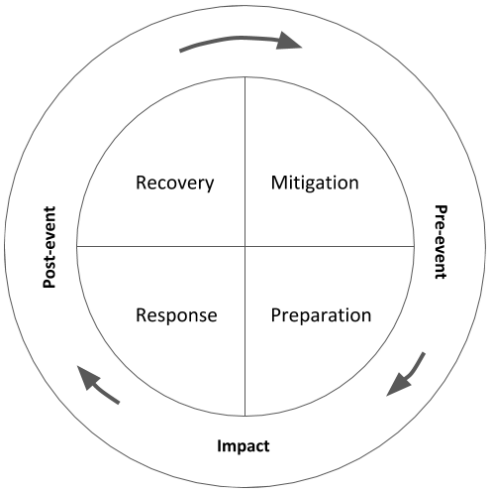


Figure 2.1: The disaster management cycle. Adapted from [15].

It is complex to make an accurate, unbiased, and useful building damage assessment. Hence, it should always be considered what the purpose of the assessment is and if the assessment captures this purpose [5]. Though many building damage assessments are made, they often do not take into account actual user needs. This results in an overflow of information, that is not put to use [17]. To our knowledge, the literature lacks a good overview of when an assessment is needed for which action and what information that assessment should entail. The only research found on actual user needs of building damage assessment is by Loos et. al. [17]. We elaborate upon this work by focusing on response actions and looking at the other information needs. To identify the different actions for which a damage assessment could be useful, the steps from *Habitat For Humanity*'s *Pathway to Permanence* were taken as a base [18]. These actions were then complemented with other actions found in literature [17, 19, 20, 21].

Table 2.1 shows the results of compiling the actions, timeline, and needed information, ordered by the timing of the actions. It is very important to note here that this overview is only made based on limited expert knowledge and a non-exhaustive literature review. Thus, before using the table as a guideline it should be studied further together with users. Moreover, it is likely that there is no single truth and that the needed information and timeline highly depends on the specific situation.

Once the response action(s) and associated requirements are identified, the most suitable source of data should be chosen. There are different sources of data that can be used to create a building damage assessment. The main distinction is between field surveys, further divided into household surveys and streetview imagery, and sensed data, further divided into unmanned aerial vehicle (UAV) and satellite imagery. More sources of sensed data, such as data from aircrafts exist, but they are left out of this research. Which data source is the preferred one depends on many factors, as illustrated in Table 2.2. The categories in this table were loosely inspired by previous work on characterizing data ecosystems [22] and extended with specific damage assessment relevant factors. Again, this is a start of research that should be further verified with practitioners and again the pros and cons highly depend on the situation. For example, in general, it can be assumed that satellite data can be retrieved faster than a household survey, but this depends largely on the available infrastructure.

As UAVs became more accessible and satellite resolution became higher, damage assessment from those sources has become more popular over the years [23]. The main disadvantages of using aerial imagery are the limited resolution and that not all damage is visible on the imagery due to the quasi-vertical nature. Examples of

Response action	Needed information	Disaster management phase	Damage scale requirements	Other data requirements
Search and rescue	which buildings should be investigated	very urgent response	destroyed vs not destroyed	building structure & usage
Tents or transitional shelter	which households and institutions need tents or transitional shelter	urgent response	destroyed vs not destroyed	building usage
Emergency shelter kits	which buildings need to be rebuilt	response	several granularities	building structure & usage
Cash assistance	which households have damaged property and are vulnerable	early recovery	several granularities	vulnerability, coping capacity, building usage, recovery costs
Request for external assistance	extent and severity of damage	response and recovery	several granularities	vulnerability, coping capacity, building usage, recovery costs
Monitor process	which buildings have changed over the given period	recovery and mitigation	several granularities	building usage

Table 2.1: Different response actions after a disaster for which building damage assessments can provide essential information. This is a rough sketch made by the author and may not reflect the actual situation.

the latter are cracks in walls or water damage after the water has receded. This results mainly in difficulty to correctly detect intermediate classes [24]. While many argue that damage assessment from sensed data can be useful, and is often used as a source to create damage assessments[1], it remains unclear if it gives high enough quality and if so, for which actions [17]. Some research has been done on the alignment between damage labels from satellite imagery and household surveys, with differing results. For example, in [25] it was shown that the agreement of the remote sensed and household survey damage labels was solely 36 percent, mainly attributing this disappointing number to the low resolution of the satellite imagery, while [24] summarizes several papers that have a 60-70 percent overlap between aerial and field survey labels. Both those articles do however not dive into the question of what the usefulness of the labels from both data sources was. For the Post-Disaster Needs Assessment (PDNA) after the earthquake that struck Haiti in 2010 [26], the damage assessment was done based on aerial imagery. A study was done to compare those labels with labels gathered by a field survey. They found that there was a 61% overlap when differentiating four classes and 86% for two classes [27]. Nevertheless, the aerial labels were used and thus apparently yielded good enough results for their purpose. This is the only study found that clearly defines the usage of the aerial annotations and compares those labels to ground data.

Due to the different results and the lack of knowledge of actual use of the data, it is still an open research question for which response actions remotely sensed damage assessment can give the required information. Nevertheless, since many damage maps are being produced from satellite imagery, there seems to be added value in this source of data.

When using remotely sensed data, it should be decided if the damage assessment is done manually or automated. Many approaches have been developed for automated damage assessment, but so far they have not been put to operation [28]. Whether a manual or automated assessment should be favored depends on several

1 For example by https://unitar.org/maps/latest-maps and https://www.hotosm.org/

	Dimension	Remote sensing		Field survey	
		Satellite	UAV	Streetview	Household survey
Quality	Granularity of damage	−−	−	+	++
	Spatial coverage	++	+	−	−−
	Elaborateness of data	−−	−−	−	++
Acquisition	Relief of workload	++	+	−	−−
	Degree of access	++	+	−	−
	Weather independence	−−	−	+	+
	Timeliness	++	+	−	−−

Table 2.2: The pros (+) and cons (−) of different data sources from which building damage assessment can be done.

factors. An attempt is made to compile the most important factors and categorize them into the three IDEO circles: desirability, feasibility, and viability [29] in Figure 2.2. This framework was chosen since the method of assessment must be realistic in all those three categories, only then it can make a sustainable impact. The identified factors belonging to each category are shown in Table 2.3. These factors are given from the perspective of the organization producing the damage assessment. Again this is a first attempt and should be designed further with experts.

Whether each factor is favorable in a manual or automated aerial assessment depends greatly on the situation, thus it was decided not to rate them here. For example, if looking at *capacity to use the methods* then for 510 there is capacity available for the automated method, whereas in organizations that have been doing manual assessments for years, there is likely more capacity for those. Thus, the factors and categorization guide as a framework on which any aerial assessment method can be scored. Besides improving the models used for automated damage assessment, this research mainly focuses on getting a clearer picture of the *certainty of quality* of these models, which in turn will help the *psychological acceptance*.

To conclude, this section gave the first attempt to map the complexity of factors that influence the value of a building damage assessment. Since such an assessment is a complex and still undiscovered process, it should be emphasized again that this research only is the first attempt, and a good co-design process together with users should be carried out before any conclusions can be made. This process is ongoing

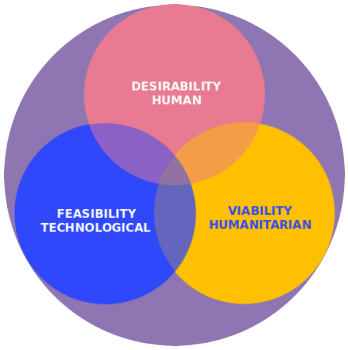


Figure 2.2: The three circles of innovation by IDEO [29] and adapted by 510.

Category	Factor
Desirability	fit with response action
	timeliness
	psychological acceptance
	objectivity
	certainty of quality
Viability	workload
	sustainability in the long run
Feasibility	capacity to use methods available
	management needed

Table 2.3: Factors a building damage assessment method from aerial imagery should be assessed on.

at the moment within 510. Moreover, the whole discussion in this section only touches upon a small part of the whole ecosystem and needs. For example, how and to whom certain information should be communicated is yet another question.

2.3 DAMAGE SCALES

How to classify damage has been a constant struggle in the field of damage assessment. The needed number of damage levels and their categorization depend on the purpose of the assessment, the type of natural hazard, and the data source. Many different scales have been developed over the years. An example of a commonly used scale is the EMS-98 for seismic damage [30]. This scale differentiates five levels of damage caused by earthquakes. However, when using aerial imagery as the data source, many damage scales are not applicable since not all types of damage in the traditional damage scales can be detected from space [31, 32].

This is why some have recently attempted to develop a scale for space imagery [31, 33]. One of them is the *Joint Damage Scale* which has the specific purpose to be generalized across disaster types and geographical locations, while still being relevant for the needs in operations [33]. The scale is depicted in Figure 2.3. It is important to note that choosing the most suitable scale is again a process that should be done together with the users, not solely by an engineer.

Damage level	Structure description
No damage	Undisturbed. No sign of water, structural or shingle damage, or burn marks
Minor damage	Building partially burnt, water surrounding structure, volcanic flow nearby, roof elements missing, or visible cracks
Major damage	Partial wall or roof collapse, encroaching volcanic flow, or surrounded by water/mud
Destroyed	Scorched, completely collapsed, partially/completely covered with water/mud, or no longer present

Figure 2.3: The damage categories as defined by the Joint Damage Scale. Figure adapted from [33].

Section summary

- Little is known about the needs of the users of a building damage assessment
- The requirements of a building damage assessment depend on the response action
- The suitability of the source of data depends on those requirements
- Satellite imagery is used as a source of data, where damage is annotated by humans
- Automated assessment from satellite imagery is nowadays not used in practice while, judging from the advantages, there is a lot of potential

2.4 REMOTE SENSING

Remote sensing is *the practice of deriving information about the Earth's land and water surfaces using images acquired from an overhead perspective, using electromagnetic radiation in one or more regions of the electromagnetic spectrum, reflected or emitted from the Earth's surface* [34].

Detection and classification of objects through remote sensing devices have become popular applications. Especially improved satellites and drones have enabled quality of data that has never been reached before. The quality and usefulness of the images depend on the characteristics of the device, as well as the characteristics of the specific image. This research solely works with data acquired by satellites and thus we look at those characteristics, though most also apply to imagery from other remote sensing devices.

The usefulness of different satellites can roughly be quantified by the three main dimensions of resolution: temporal, spatial, and spectral resolution.

- Temporal resolution indicates the revisit time of the satellite, i.e. how long does it take before the satellite covers the same location again. For a damage assessment, the optimal revisit time is as small as possible in order to have an accurate picture of the situation. The maximum revisit time depends on the purpose of the assessment, but optimally the revisit time is not more than six days.

- Spatial resolution indicates how much ground area is represented by one pixel. This is expressed in the ground sample distance (GSD). For example, a GSD of 10 meters means that one pixel represents an area of 10 x 10 meters. There is no standard on what the spatial resolution should be to be able to detect damage to buildings. But if one wants 20 pixels of both the width and height of the building and the building has a footprint of 100 m^2, then the spatial resolution should not be lower than 5 meters. When looking at the studies done on using deep learning (DL) with remote sensing imagery, many studies use imagery with a resolution of fewer than 2 meters, suggesting that ML and more specifically DL benefit from higher resolution imagery [35].

- Spectral resolution indicates which spectral bands are measured and what the wavelength width of these bands is. The light spectrum consists of a range of wavelengths and different passive sensors can capture a specific part of this spectrum. Which parts of the spectrum are captured by the satellite, and how much wavelength width is measured by each sensor defines the range and precision of the data. The most common bands are red, green, and blue, which match the human visual system. Other bands can capture relevant information as well, such as near-infrared. Another form of measuring is by using active sensors, where those sensors have their source of light and measure the amount that is illuminated back. One common active sensor is the synthetic aperture radar (SAR). This sensor can penetrate cloud cover, which can be very useful for post hazard imagery. SAR has been used for damage assessment, see Section 2.7, and proven useful, though it is also quite sensitive and gets easily distorted by changes in, for example, vegetation.

Apart from the specifics of the satellite, the appearance of an image is also influenced by environmental factors, especially by the position of the satellite and the Sun relative to the area of interest. Those positions are often captured in two numbers, the *azimuth*, and the *elevation angle*. Both the Sun and the satellite are assigned these angles, as visualized in Figure 2.4. The azimuth angle is the angle between the object (Sun or satellite) and the North axis, measured clockwise on the horizontal plane. The elevation angle is the vertical angle between the object and the horizontal plane.

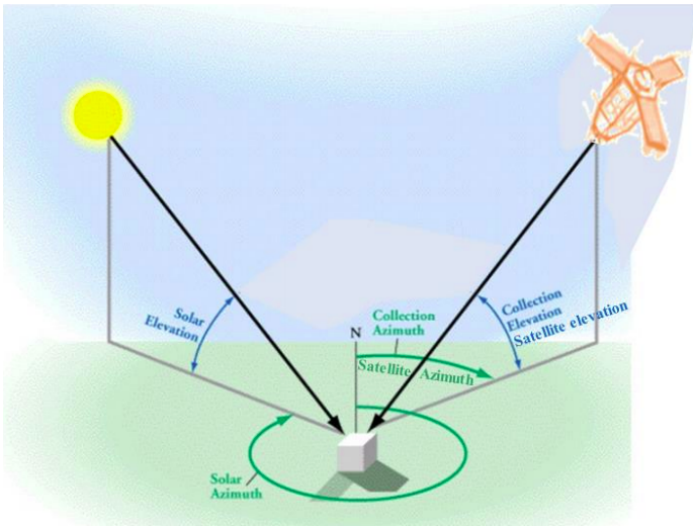


Figure 2.4: The different angles of the Sun (i.e. Solar) and satellite in remote sensing. Adapted from [36].

The elevation angle of the satellite can also be expressed as the *off-nadir angle* where off-nadir angle $= 90° -$ elevation angle. A larger off-nadir angle, and thus a lower elevation angle, means that objects are shown more from the side which makes them look different. Especially if two images of the same area are taken from different off-nadir angles, this can give difficulty in recognizing the same object. How much the object gets shifted, largely depends on the tallness of the object. Moreover, a higher off-nadir angle degrades the resolution of the imagery as larger patches of area are shown by a smaller number of pixels. The elevation angle of the Sun is mainly an indication of the brightness of the image and the length of the shadows. The azimuth of the satellite and the Sun is mainly an indication of the direction and distortion of buildings and shadows. Two images with the same off-nadir angle but a 180-degree difference in the target azimuth can appear very different and thus cause trouble for the model.

2.5 SOURCES OF SATELLITE IMAGERY

While we cannot influence the quality of the available imagery, we can make an educated choice on which satellites to use as the data source, based on the three main types of resolution as explained in the previous section. Before assessing the resolutions, in practice, two other requirements are that the satellite imagery should cover the impacted area and be freely accessible to 510.

Several initiatives provide free satellite imagery with global coverage, of which the most famous ones are Landsat[2] and Sentinel[3]. Landsat is a program of NASA and U.S. Geological Survey (USGS), which consists of a constellation of satellites. Sentinel is a constellation of satellites of the European Space Agency (ESA). Initiatives also started to pop up that are especially focused on providing imagery and information during and after a disaster, of which an important one is the International Charter for Space and Major Disasters[4]. A group of satellite imagery providers is affiliated with this charter and make their imagery available for free after disasters. One of those affiliated partners is the Maxar/DigitalGlobe Open

2 https://landsat.gsfc.nasa.gov/
3 https://sentinel.esa.int/
4 https://disasterscharter.org/

Source	Spatial (m)	Temporal (days)	Spectral (bands)
Landsat (7&8)	30	16	11
Sentinel (1&2)	10	6-10	13
Maxar	0.25-0.6	1-3	4-28

Table 2.4: Resolutions of different satellite initiatives. The shown specifications of the satellites owned by Maxar are from the satellites GeoEye1, Quikcbird, WorldView2, and WorldView3. Numbers retrieved from [37, 38].

Data Program, from now on referred to as *Maxar*, which seems most consistent in providing imagery both through the charter as well as on their own website[5].

In Table 2.4 the resolutions of Landsat 7&8, Sentinel 1&2, and Maxar are compared. For the Maxar comparison, the satellites that seem to appear most often in their open data were taken into account which are GeoEye1, Quikcbird, WorldView2, and WorldView3. As can be seen, the spatial and temporal resolution of the Maxar data is better compared to Sentinel and Landsat, whereas the spectral resolution depends on the satellite providing the data. The mentioned spatial resolutions of Maxar are of the panchromatic band, the resolutions of the multi-spectral bands range from 1 to 2.4 meters. Nonetheless, the resolutions of the multi-spectral bands can be increased to the ones of the panchromatic band through a technique named pan sharpening.

The 30-meter pixel size of Landsat is large for building damage assessment. Research has been done on using Landsat for building detection, but those researches used tricks to improve the resolution or used information from neighboring pixels [39, 40]. In combination with the long revisit time, Landsat does not seem suitable for the application of this research. One interesting characteristic of the Sentinel satellites is that Sentinel-1 has a SAR band.

To limit the scope of this research and the requirements of available data, it was chosen to only use the bands from the visible spectrum, i.e. panchromatic, red, green, and blue bands. Since these are provided by all the satellites of Maxar and this source has the best spatial and temporal resolution, this data source was chosen to be used for this research. A disadvantage for its usage in practice is, however, that Maxar decides which data is made openly available and when. This has a result that data might not be available for all disasters and might have a delay.

2.6 GROUND TRUTH DATA

In addition to satellite imagery, ground truth data is needed to train the model. The ground truth data should consist of building outlines and an indication of the level of damage for each of those outlines. Previous research has created their datasets from different sources, but they are not open-source [41], only cover a small sample of disasters [42], or a limited geographical area [43]. Recently, a new dataset named xBD was published which is the first of its kind [33]. It was decided to use this as a source of ground truth data for this research since it is the dataset with the widest coverage of locations and types of disasters.

The ground truth data of xBD consists of building outlines, damage labels, and other metadata. The building outlines are coupled to matching Maxar data and this Maxar data is provided within the dataset as PNGs of $1,024 \times 1,024$ pixels. Because of the ease of use, those images are used in this research. In practice, the original TIFF files can be easily downloaded from the Maxar website and used as input. Moreover, since the building outlines are georeferenced, other sources of satellite imagery could be used if extra precaution to alignment is taken. The xBD dataset uses the Joint Damage Scale which differentiates four classes, *no damage*, *minor dam-*

5 https://www.digitalglobe.com/ecosystem/open-data

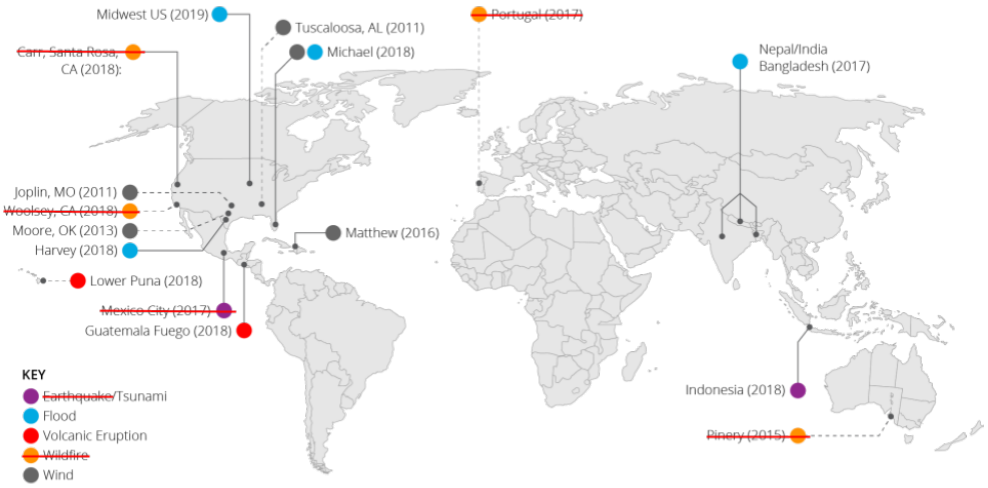


Figure 2.5: The location and type of the different disasters in the xBD dataset. The crossed out disasters will not be used in this research. Amended from [33].

age, *major damage*, and *destroyed*, as further described in 2.3. The damage class labels and building polygons in this dataset were assigned from the satellite imagery by human annotators. All polygons were drawn on the pre imagery, which might result in misalignment on the post imagery, especially if off-nadir angles differ. The damage classification was done by annotators and even though an extensive review process was in place, the damage classification remains subjective and some damages are hard to see well due to the resolution.

The dataset covers 19 disasters of 5 different types, as shown in Figure 2.5. The dataset that was released and used during this research contains 316,114 unique buildings given as outline polygons. At the end of this research, another set has been released by which the total amount of buildings in the xBD dataset was increased to 425,368. This additional set has not been used during this research due to time constraints. No previous research has been published with the xBD dataset, so no direct comparison of performance can be made.

The creators of the xBD dataset aim to represent a wide range of geographical locations and types of disasters. Though the largest part of the data is still in the USA, to our knowledge it is more diverse and larger than any current dataset on building damage, and thus creates an interesting ground for this research. The labels are directly coupled to the Maxar data, so this is another argument to make use of the xBD dataset since it is certain that the labels align with the images.

Though it is very valuable that the xBD dataset exists now, it is not perfect. A few critical notes that should be considered are:

- xBD only includes disasters for which data could be collected with all needed requirements. For example, high image quality was demanded which included that only very little clouds should be present. This means that in practice data with similar quality might not be available in time for all disasters.
- Though more diverse than other available datasets, the range of geographical locations is still rather limited. 11 out of 19 are located in the US, none of them are located in South America or Africa.

2.7 STATE OF THE ART IN AUTOMATIC BUILDING DAMAGE CLASSIFICATION

Automatic building damage classification using remotely sensed information has gotten increased attention during the last years. Different image sources with different resolutions are being used, where the main focus is on imagery obtained from satellites and drones. This research solely focuses on imagery from satellites, but many of the techniques can be applied to both source types [41].

The first research on automated damage assessment mainly explored the use of geographic information system (GIS) techniques. This showed that it is possible to automatically detect damage from aerial imagery, but results in performance were not always satisfying [28]. Mainly models using radar SAR data have shown to work well, by using relatively simple techniques of change detection between pre and post imagery [44, 45, 46]. However, radar data is openly available only with low resolution, and thus no assessments with this data can be made on building level [32], which is the focus of this research.

Due to the complex nature of features indicating damage and with the increased popularity of machine learning (ML), research started exploring the use of ML techniques for automatic classification, especially focusing on convolutional neural network (CNN)s. Though relatively little comparison work has been done, CNN features have been shown to outperform textural features [47]. In previous research, the use of CNNs has shown promising results, but at the same time no models using these novel technologies have yet been applied in operational conditions [28]. Hence, this research tries to improve those models while focusing on the criteria of applicability across disasters, without labeled data of the target disaster.

Table 2.5 shows a non-exhaustive overview of the research done on automated damage classification with satellite imagery using neural networks on a spatial granularity scale of buildings. For each study, the best performing model in the study when trained and tested on the same disaster is shown. The results are shown in accuracy since this was the only measure reported across all articles. Sadly, accuracy is not a very good comparison method here because of the class imbalances. Nonetheless, from these accuracies, it can be concluded that all models except [48] are learning damage related features since their accuracy is larger than that of a naive classifier that always predicts the majority class. The quality of the models from the different articles cannot be compared based on the reported accuracies since there are more differences between the studies than solely the model. These include class balance, disaster type, and data sources.

As can be seen from the table, most articles use a CNN as a basis and might add extra elements to it. Moreover, as time progressed the models being used have become more complex while there is no direct correlation with an increase in accuracy. A decision that also influences the model is whether to use pre and post imagery or only post imagery. As can be seen from the table, this choice differs per study. Only using post imagery lowers the image requirements, but it might lead to worse performance. Two of the papers did a comparison, and though with different margins, they showed that including pre imagery never harms the performance [42, 49], but whether it increases depends on the disaster and the model. What the improvement is of using pre imagery on a wider range of disasters and if that outweighs the extra data requirement, has to be researched further.

All studies solely separate two classes, making a distinction between *destroyed* and *not destroyed*. Not classifying more than two classes has been explained by the statement that other levels of damage are hard to distinguish from satellite imagery [32, 28]. Whether this is true has to be tested. Moreover, the damage classes should fit the purpose of the assessment and while binary labels can in some cases be enough, it is highly likely that a more precise distinction is needed for other purposes.

Reference & Publication date	Accuracy	Model	Disaster type	Satellite data			Labeled data			Output	
				Source	Time	Bands	Source	Type	Label perc. (destr/not)	Damage granularity	Spatial granularity
[48] 11-2016	74.1	FNN (2 layers)	earthquake	Maxar	pre & post	panchromatic, RGB, near infrared	UNITAR/UNOSAT	remote sensed	11/89	binary	pixel
[49] 05-2017	95.6	Siamese CNN (4 layers)	tsunami	PASCO	pre & post	RGB	government	field survey	40/60	binary	building
[50] 05-2019	97.3	CNN (4 layers)	hurricane	Maxar	post	RGB	Tomnod	remote sensed	50/50	binary	building
[47] 05-2019	87.6	CNN (3 layers) + Random Forest	earthquake	Maxar	pre & post	RGB	UNITAR/UNOSAT	remote sensed	36/64	binary	building
[42] 10-2019	78.0	Siamese CNN plus CNN after concatenation	earthquake	Maxar	pre & post	RGB	UNITAR/UNOSAT	remote sensed	55/45	binary	building
[41] 11-2019	87.1	CNN (adaptation of densenet121)	earthquake	Maxar	post	RGB	authors	remote sensed	50/50	binary	patch of 120x120 pixels
[51] 01-2020	88.8	CNN (VGG16)	earthquake	Maxar	pre & post	RGB	UNITAR/UNOSAT	remote sensed	34/66	binary	building

Table 2.5: Overview of research done on automated building damage assessment from satellite imagery using neural networks. *Maxar* refers to the Maxar/DigitalGlobe Open Data Program

As source of the damage labels, most studies use the analysis done by UNITAR/UNOSAT. This data has been created through annotation of satellite imagery by professional analysts and is openly available through the Humanitarian Data Exchange (HDX)[6]. The original UNOSAT assessments used in the cited studies differentiate five damage levels, but all cited studies binarize those five levels to two. Three out of four studies that use the UNOSAT data, group all classes other than *destroyed* to one class, while [42] also include the level of *severe damage* in the damaged class. The class consisting of *destroyed* and possibly *severe damage* buildings is from here on referred to as *destroyed* and the other class as *not destroyed*. When inspecting the sources of satellite data, all except one use data from the Maxar/DigitalGlobe Open Data Program, which is the same source as this research uses. This seems to be the data that is openly available after disasters with the highest resolution.

Analyzing the class balance, most studies work with an imbalanced dataset. The two studies that have a balanced dataset prepared this on purpose, and thus this is not the original data. This applies to more of the studies, where solely a part of the data was selected. This is not a realistic scenario when testing on a current disaster, and thus it is likely that in real case scenarios the class imbalance is even larger. What the effect is of the class imbalance is hard to assess from previous research. The study with the worst imbalance gains the lowest performance, but this low result in performance is likely to also correlate with the model architecture. Simultaneously, it could be argued that a larger imbalance improves performance when measured in terms of accuracy since the model tends to lean towards the majority class. To know if the model indeed shows this behavior, the separate performance of both classes should be analyzed. Solely two papers report this performance, and in both cases the model does perform better on the majority class (*not destroyed*) than the minority (*destroyed*) class [47, 51]. Thus, those cases show that a larger class imbalance leads to better accuracy. However, this higher accuracy does not have to indicate a better predictive value when the class imbalance is larger. Thus, the accuracy and class distribution should always be considered jointly before jumping to conclusions.

Regarding the type of damage, all studies focus on one type. Combined they cover three types: *earthquake, tsunami*, and *hurricane*, where earthquake is by far the most researched type. From solely judging the accuracies, earthquake damage seems to be harder to classify. However, due to the differentiation of more factors than just the disaster type, it should be further researched if indeed disaster type influences the performance.

All results shown in Table 2.5 have been trained and tested on the same disaster, but two of the studies also experimented with training on a set of disasters and testing on another disaster [41, 42], where both studies focus on earthquake damage but use a mix of disasters with different geographical locations. Both papers found that the performance drops when not including the test disaster in the training set. The set-up of [41] has seven disasters in total. Every experiment is trained on six of them and then tested on the one that is left out. This is repeated for four out of the seven disasters. They show that the performance of the test disaster significantly differs per disaster. They attribute this difference to differences in image quality and building topology, but do not go on to quantify this. The other paper [42] has a total of three disasters. In this study, the model is trained on one or two disasters and then tested on the third. This is done for two out of the three disasters. They show again that there is a drop in performance when not training on the test disaster, reaching 85% and 92% of the original performance, which is a surprisingly small gap. These results are reached when training on two disasters, while training on one disaster gives a lower performance, and thus they show that adding more disasters to the training dataset is beneficial. Moreover, similar to [41], they find that the performance differs per test disaster. They show that this difference in performance between disasters also holds when the model is trained on the same disaster it is

6 https://data.humdata.org

tested on. Both papers also apply fine-tuning as an experiment. In [42] this was 10% of the available buildings and in [41] between 15% and 25%. They show that the fine-tuning improves performance in all set-ups. However, it is questionable if having this percentage of labeled data in a real situation is realistic.

Section summary

- The usage of convolutional neural network (CNN)s for automated building damage assessment has shown promising results
- All previous research used binary labels and tested the model on one damage type
- One study has applied the same model on several disasters (of the same damage type) and showed differences in performance between disasters
- Two studies experimented with testing on a disaster the model has not been trained on and showed a drop in performance in all cases, but depending on the disaster this drop was relatively small [41, 42]. Moreover, they both showed a difference in the magnitude of this drop per disaster

Chapter summary

- A natural hazard together with the vulnerability and coping capacity of a community define a disaster
- User needs of automated building damage assessment are important to take into account for the design of a tool
- The most suitable satellite data source for this research is the Maxar/DigitalGlobe Open Data Program
- The most suitable ground-truth data that covers a wide range of disaster is the xBD dataset
- Previous research with automated assessment by a convolutional neural network (CNN) has been done, but these studies were limited in researching the differences between disasters, the number of classes of damage, and transferability to a disaster the model has not been trained on

3 RESEARCH METHODOLOGY

The field of machine learning (ML) builds models whose values of parameters are decided by the model itself. This is done during a learning process which is fueled by data. Machine learning (ML) has been applied to a wide range of problems. The problems it tries to solve have become more complex in nature, and thus to solve them methods are being developed that can capture these complex, often non-linear, relationships. This chapter explores the different elements needed to create and evaluate a ML model, especially focusing on a subfield of ML named deep learning (DL). The chapter looks at different layers, loss functions, activation functions, optimization methods, regularization techniques, and architectures. After which it explores the concepts of transfer learning, imbalance, and performance measures.

3.1 FORMALIZING THE PROBLEM

Every ML problem can be defined by three main components. An excellent more elaborate description of these components can be found in [52]. Summarized they amount to:

- The data, which consists of the feature space, $\mathcal{X}$, and the label space, $\mathcal{Y}$.
- The hypothesis space, $\mathcal{H}$, which defines all possible mappings from the feature space to the label space, and is determined by the choice of the model.
- The loss function, which defines how the quality of a particular set of weights is measured.

In this chapter different options of all these components in the context of deep learning (DL) will be discussed. In Section 5.5, these characterizations are specified for the specific model used in this research.

3.2 DEEP LEARNING

The most popular subfield of ML for capturing complex relations between input and output is currently DL. In DL so-called neural networks with several layers are used. Those layers contain weights and the values of those weights are adjusted during the train process by calculating the error in terms of a *loss function*, after which this error is backpropagated through all layers by the technique named *backpropagation*. These deep networks are good at capturing complex non-linear relations. This research solely focuses on DL models and thus this section lays out some of the most important blocks of those models. Since in every block there are many varieties possible, this document focuses on a few, specifically highlighting the choices of those blocks that are used in this research.

3.2.1 Input & output

Every model takes an input and returns an output. As an example, the simplest version of a neural network is shown in Figure 3.1. Here it can be seen that inputs are

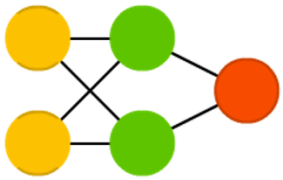


Figure 3.1: A simple neural network. Yellow indicates the inputs, green a layer which does a transformation of the input, and red the output. Adapted from [53].

given, after which some transformation is done to them, and thereafter an output is returned.

The range of inputs is defined by the feature space, $\mathcal{X}$, and the range of outputs by the label space, $\mathcal{Y}$. As input a wide variety of variables can be given, but mostly they are real-numbered values. They can be very concrete variables such as the *temperature*, but also variables with a less-precise meaning, such as the pixel values of an image.

The type of output of most models can be divided into two categories: classification and regression. With regression, a continuous output is given which results in an output size of one. This output can range from $-\infty$ to ∞ while it is commonly standardized to range from zero to one, in which case the label space is $\mathcal{Y} = [0, 1]$. With classification a discrete output is given, where the output often indicates the probability of the data point belonging to each class, thus the output size equals the number of classes, N, and the label space is $\mathcal{Y} = \{C_1, \ldots, C_N\}$.

Which output type should be chosen highly depends on the problem at hand. In the case of automated building damage assessment, both could be used. Classification fits the data neatly since the damage scale comes as discrete numbers. At the same time, one could argue that regression is a better fit since damage is a continuous scale and thus the classes are monotonically related to each other.

For this research, it was chosen to stick to classification since this is the most common method. However, future research could experiment with regression vs classification. The remainder of this chapter focuses on blocks that can be used for classification. Some of the blocks discussed apply to both classification and regression, such as most layers, others are specific to classification, such as the loss.

3.2.2 Layers

A neural network consists of a chain of layers. These layers transform the input according to certain formulas. This research focuses on a type of architecture named convolutional neural network (CNN). Figure 3.2 shows a schematic overview of a typical structure of the CNN. The input is followed by a set of convolutional layers, combined with pool layers, and thereafter one or more fully connected layers follow before returning the output.

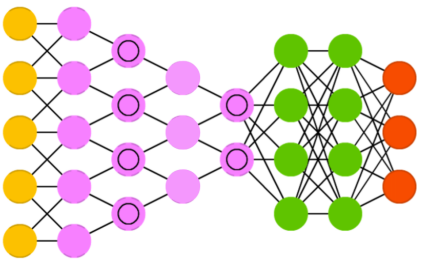


Figure 3.2: A typical convolutional neural network. Yellow indicates the inputs, purple convolutional layers, the circles withing the purple circles pooling layers, green fully connected layers, and red the output. Adapted from [53].

3.2.2.1 *Fully connected layer*

While often occurring at the end of a CNN, the fully connected layer is the most straightforward layer. A fully connected layer consists of neurons. Each of these neurons is connected with all outputs of the previous layer. Each connection contains a weight, and those weights are learned by the network. Mathematically, given an input $\mathbf{x}$, a fully connected layer outputs $\mathbf{y}$ according to

$$\mathbf{y} = \mathbf{W} \cdot \mathbf{x} + \mathbf{b} \,, \tag{3.1}$$

where $\mathbf{W}$ is a matrix containing the weights and $\mathbf{b}$ the bias vector.

3.2.2.2 *Convolutional layer*

When working with raw image data, solely using fully connected layers is not a good choice. The number of neurons explodes quickly and the chance of overfitting is high. Say we have a relatively small input image with a width and height of 32 pixels and containing the three RGB channels. This picture would consist of $32 \times 32 \times 3$ pixels. Say the fully connected layer consists of 500 neurons. This would result in $32 \cdot 32 \cdot 3 \cdot 500 = 1.5$ million weights for one layer.

To minimize the number of weights and to circumvent overfitting, the convolutional layer was invented, which takes advantage of the properties of images. In images, it is often about recognizing certain patterns, that are formed by a small region of the input. Thus, a convolutional layer works with neurons that are connected to only a small region. The region they are connected to is called the receptive field. The receptive field indicates the width and height of the neuron, the neuron always acts over the full depth. So given an input of $32 \times 32 \times 3$ and a neuron with a receptive field of 5×5, this neuron will have $5 \cdot 5 \cdot 3 = 75$ weights. This set of 75 weights is called the filter. The depth of the output can be increased by simply adding more channels of filters, each containing their own weights.

How many neurons each filter has depends on the size of the input, and on two hyperparameters: the stride and the padding. The stride indicates how big the steps are with which we slide the filter. When the stride is one, the filter is slid pixel by pixel. When the stride is two, it skips one pixel after every slide, which results in smaller outputs spatially. The other factor is padding. The input can be padded with zeroes around the borders, where a padding of one means that one border of zeroes is added. Combining these, the output size is determined by $\frac{N-F+2P}{S+1}$ where N is the input size, F the filter size, P the padding size, and S the stride. For example, if we have a $32 \times 32 \times 3$ input, so N is 32, a F of $5 \times 5 \times 3$, a stride of 2 (S=2) and no padding (P=0), the output width and height would be $\frac{32-5}{2+1} = 9$. This will give an output of size $9 \times 9 \times 1$.

Say we use the settings of the previous example, and 10 channels of filters, this would result in $9 \cdot 9 \cdot 10 = 810$ neurons where each neuron has $5 \cdot 5 \cdot 3 = 75$ weights, totaling $810 \cdot 75 = 60750$ weights. This number grows very quickly when working with larger inputs and more channels of filters. Thus the concept of *parameter sharing* is almost always used in a convolutional layer. Using the notion that important features, such as edges, look similar across the image, the weights between the neurons over different parts of the image are shared. This means that there is only one set of weights for each channel, and thus in the example results in $75 \cdot 10 = 750$ weights for the whole layer. Another advantage of this smaller set of weights is that it helps to prevent overfitting.

3.2.2.3 *Pooling layer*

A pooling layer is often added between convolutional layers. The goal of the pooling layer is to reduce the feature map size and thus reduce the number of weights and computation time for consecutive layers. This again reduces overfitting. A pooling layer has no parameters, it does have a kernel with a kernel size and a

stride. Over the receptive field, it applies a transformation which is commonly taking the maximum or average of the values in the receptive field. The kernel size and stride determine what the size is of the output, according to the same formula as in a convolutional layer, but without the pooling, i.e. $\frac{N-F}{S+1}$.

3.2.3 Activation function

After each convolutional and fully connected layer, an activation function is applied. Through this activation function, non-linearity is introduced to the network. This is essential for a neural network since the whole advantage of a neural network is that it can learn complex non-linear functions. Without activation functions, it would continue being a chain of dot products and hence could also be replaced by linear regression. The four main functions used as activation functions are sigmoid, tanh, ReLU, and Leaky ReLU. See Figure 3.3 for an overview of the shapes of those functions.

Nowadays, sigmoid and tanh are not often used anymore because they experience the so-called vanishing gradient problem. This means that the gradients of weights become small, which causes the weights not to change and thus the network not to learn. To solve this problem, ReLU was invented [54]. ReLU leaves all input larger than zero as they are, and clamps all inputs below zero to zero. This also makes the network easier to train, since the outputs are more sparse. A problem of ReLU is that it can cause dead neurons, i.e. neurons that always output zero after the ReLU operation, no matter the input. To tackle this, Leaky ReLU was invented, where inputs below zero are multiplied with a small value to get the output.

(Leaky) ReLU is only suited as the activation function for hidden layers, not for the output layer. For multi-class classification the standard activation function for the output layer is softmax. Softmax squeezes the input to the $[0, 1]$ range and results in the sum of outputs being 1, and thus these outputs can be interpreted as probabilities. The formula of softmax is

$$\text{softmax}(x)_i = \frac{exp(x_i)}{\sum_j exp(x_j)} \ . \tag{3.2}$$

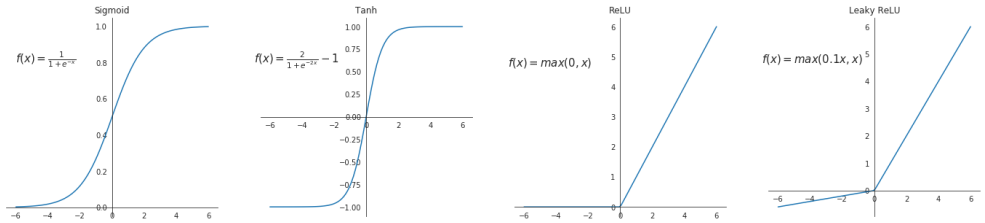


Figure 3.3: Plots and formulas of the four most used activation functions.

3.2.4 Loss function

The loss function defines how wrong predictions are penalized, which influences how the weights are updated during error backpropagation. Several loss functions for multiclass classification exist. The most popular one of them is the cross-entropy loss, which is also used throughout this research. Defining $y_{ic} \in \{0, 1\}$ as the label for data point i in class c and $\hat{y}_{ic} \in [0, 1]$ as the predicted probability of data point i belonging to class c, the cross-entropy loss is defined as

$$J = -\frac{1}{N} \sum_{i=1}^{N} \sum_{c=1}^{C} y_{ic} \log(\hat{y}_{ic}) + (1 - y_{ic}) \log(1 - \hat{y}_{ic}) \ . \tag{3.3}$$

3.2.5 Optimization

The final goal of any ML algorithm is to find the set of weights such that the loss function is minimized. How the weights are changed depends on the chosen optimization method. Most practical optimizations methods use the gradient of the loss function with respect to the weights of the network as part of the update formula. This gradient, further on indicated as $\nabla_{\mathbf{w}} J(\mathbf{w}_t; \mathbf{X}_i, \mathbf{y}_i)$, is calculated by a technique called *backpropagation*. Backpropagation starts with the loss of the network and then computes the derivatives for all weights, i.e. the gradient, of all layers in the network by using the chain rule. Over the years many optimization methods have been developed, of which the most used ones are explained below.

3.2.5.1 *Stochastic Gradient Descent*

We can optimize the loss by moving along the direction where the loss is negative. With this approach, we always know that the loss will decrease with an infinitely small step along the gradient direction. We can thus update the weights according to

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \alpha \nabla_{\mathbf{w}} J(\mathbf{w}_t; \mathbf{X}_i, \mathbf{y}_i) \ , \tag{3.4}$$

where $\nabla_{\mathbf{w}} J(\mathbf{w}_t; \mathbf{X}_i, \mathbf{y}_i)$ is the gradient of the loss function with respect to the weights and α is the learning rate which is a hyperparameter that defines how large the step towards the gradient is. A too-large α can cause missing the optimum, while a too low α can cause very slow convergence.

This optimization method is named *gradient descent*. Gradient descent is computed over the full dataset. This is very inefficient since often many iterations are needed to converge towards the optimum and computing the gradient over the whole dataset is expensive. For this reason, stochastic gradient descent (SGD) was invented, where the gradient is computed per sample. This, in turn, has the problem that the gradient, and thus the weight updates can be very noisy. As a compromise, the status quo is now the *batch gradient descent*, where the gradient is computed over a subset, i.e. batch, of the training samples. The batch size is a hyperparameter and its common values are 32, 64, or 128. Though officially named batch gradient descent, this is nowadays often referred to as stochastic gradient descent (SGD).

There are two big problems with SGD: 1) it tends to overshoot in directions with steep gradients, while slowly converging on the shallow gradients, 2) it can easily get stuck at local optima or saddle points since the gradient is zero at those locations.

3.2.5.2 *Momentum*

Momentum is a variation of SGD which uses the exponentially weighted average of the gradient instead of solely the gradient [55]. This can be defined as

$$\begin{aligned} \mathbf{s}_t &= \beta \mathbf{s}_{t-1} + (1-\beta) \nabla_{\mathbf{w}} J(\mathbf{w}_t) \ , \\ \mathbf{w}_{t+1} &= \mathbf{w}_t - \alpha \mathbf{s}_t \ , \end{aligned} \tag{3.5}$$

where $\mathbf{s}$ is the exponentially weighted average of the gradient. β is a hyperparameter which indicates how much of the previous steps are taken into account. As a rule of thumb, the number of previous steps that have a significant impact approximately equals $\frac{1}{1-\beta}$. β is most commonly set to 0.9, which means that about the last 10 steps have a significant weight in the update of the current step.

By taking into account previous values, smoothing occurs because less overshooting takes place in the steep directions while a big gradient is kept in the shallow directions, see Figure 3.4 for a visualization of this behavior. This causes the model to converge faster and get less stuck at local minima.

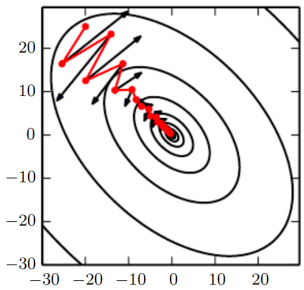


Figure 3.4: A rough sketch of how optimization with SGD (black) and momentum (red) converge to the minimum. Adapted from [56].

3.2.5.3 *RMSProp*

RMSProp, proposed by Tieleman and Hinton [57], is an optimization method that takes another approach to flatten out oscillations.

Instead of having the same learning rate for all the weights, it computes a learning rate for each weight, which is also adaptive over time. This is done by dividing the general learning rate, α, by the square root of the exponentially weighted average of the squared gradient. Weights with absolute larger derivatives have a larger exponentially weighted average of the squared gradient and thus get a smaller learning rate. The opposite is true for weights with absolute smaller derivatives. Putting it together, the algorithm can be written as

$$\begin{aligned} \mathbf{r}_t &= \rho \mathbf{r}_{t-1} + (1-\rho)\nabla_{\mathbf{w}} J(\mathbf{w}_t) \odot \nabla_{\mathbf{w}} J(\mathbf{w}_t) \ , \\ \mathbf{w}_{t+1} &= \mathbf{w}_t - \alpha \frac{\nabla_{\mathbf{w}} J(\mathbf{w}_t)}{\sqrt{\mathbf{r}_t + \epsilon}} \ , \end{aligned} \tag{3.6}$$

where $\odot$ indicates elementwise multiplication, ϵ a very small number to prevent division by zero, and ρ is a hyperparameter that indicates the scale of the exponentially weighted average, similar to β in momentum.

3.2.5.4 *Adam*

Adam is another optimization method, which was proposed by Kingma and Ba [58]. Adam takes the best of two worlds by combining RMSProp and momentum, i.e. updating the weights according to the momentum update and tuning the learning rate per weight according to RMSProp. This damps oscillations and is robust in a wide range of applications. Because of its robustness this is also the optimization method used in this research (see Section 5.4). We write the two exponentially weighted averages of momentum and RMSProp as

$$\begin{aligned} \mathbf{s}_t &= \beta \mathbf{s}_{t-1} + (1-\beta)\nabla_{\mathbf{w}} J(\mathbf{w}_t) \ , \\ \mathbf{r}_t &= \rho \mathbf{r}_{t-1} + (1-\rho)\nabla_{\mathbf{w}} J(\mathbf{w}_t) \odot \nabla_{\mathbf{w}} J(\mathbf{w}_t) \ . \end{aligned} \tag{3.7}$$

In Adam we apply bias correction to $\mathbf{s}_t$ and $\mathbf{r}_t$. At t=0, $\mathbf{s}$ and $\mathbf{r}$ are initialized to zero. This causes the exponentially weighted averages to underestimate the actual averaged values. To create better convergence, we can diminish those underestimations. This is done by dividing $\mathbf{s}_t$ and $\mathbf{r}_t$ by $1-\beta^t$ and $1-\rho^t$ respectively. Due to the power of t, these values go to one and thus only have an impact for small ts. I.e.

$$\begin{aligned} \tilde{\mathbf{s}}_t &= \frac{\mathbf{s}_{t-1}}{1-\beta^t} \ , \\ \tilde{\mathbf{r}}_t &= \frac{\mathbf{r}_{t-1}}{1-\rho^t} \ , \end{aligned} \tag{3.8}$$

where $\tilde{\mathbf{s}}_t$ and $\tilde{\mathbf{r}}_t$ are the bias-corrected terms. The recommended values of β and ρ are 0.9 and 0.99 [58] and these have been shown to work across a wide range of networks. The weight update is then done according to

$$\mathbf{w}_{t+1} = \mathbf{w}_t - \alpha \frac{\tilde{\mathbf{s}}_t}{\sqrt{\tilde{\mathbf{r}}_t + \epsilon}} \,. \tag{3.9}$$

3.2.6 Regularization

A common problem in machine learning is overfitting, the notion of performing well on the training data, but not being able to generalize to unseen data. Several regularization techniques with the goal of preventing overfitting, have been invented and some of them have become an integral part of ML models.

3.2.6.1 L_p *regularization*

One approach to prevent overfitting is by rewarding the model to have a smaller sum of weights. This causes the focus to be more on the parameters that have a predictive value. One technique to do this is to add the L_p-norm to the loss function, where the L_p-norm is defined as

$$\|\mathbf{w}\|_p = (w_1^p + w_2^p + \cdots + w_n^p)^{\frac{1}{p}} \,. \tag{3.10}$$

The loss then becomes

$$\tilde{J}(\mathbf{w}; \mathbf{X}, \mathbf{y}) = J(\mathbf{w}; \mathbf{X}, \mathbf{y}) + \alpha \|\mathbf{w}\|_p \,, \tag{3.11}$$

where α is the regularization hyperparameter. Most often the value of p is chosen as 2 or 1, which are called L2 and L1 parameter regularization, respectively. It is also often referred to as *ridge regression*/*weight decay* and *lasso regression*, respectively. Figure 3.5 gives a visual view of how L1 and L2 regularization influence the weights. As can be seen, L1 forces the weights to go more towards zero, whereas L2 gives a more smooth function which forces the weights to be as small as possible but not per se zero. In neural networks, it is more common to use the L2 regularization and α is indicated as the weight decay parameter in most frameworks.

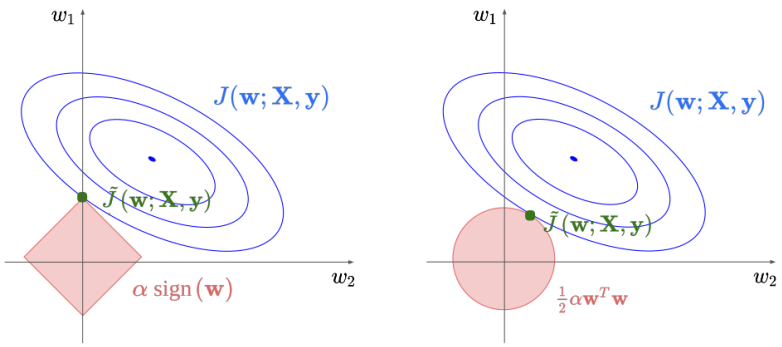
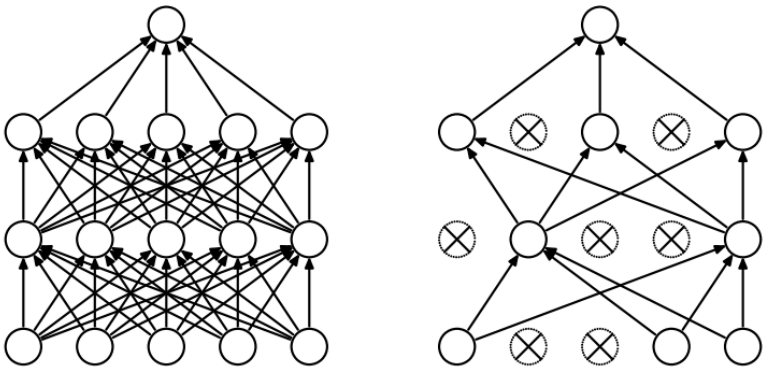


Figure 3.5: Visualization of the contours of the loss function with L1 and L2 regularization respectively. The blue contour lines indicate the same value of J, the red contour the same value of the regularization. Where the two meet is the optimum value of $\tilde{J}$. Inspired by [59].

3.2.6.2 *Dropout*

Another regularization technique is *dropout* and this technique is adopted in most deep neural networks. The idea behind dropout is to randomly turn off some of the neurons [60] by setting them to zero, as illustrated in Figure 3.6. By doing so, the



(a) Standard Neural Net (b) After applying dropout.

Figure 3.6: An illustration of dropout. The crossed units have been dropped. Adapted from [60].

network can not rely on the input of always the same neurons and is thus forced to spread its attention and thereby its weights. In this way, a sort of ensemble of networks within the networks is built and it is prevented that all weights go to one neuron.

Which fraction of the neurons is set to zero during dropout is a hyperparameter. The standard value is 0.5. Dropout is computationally cheap, but does also require the model to train about twice as many epochs till convergence [60].

3.2.6.3 *Data augmentation*

In unseen data, the data points often look slightly different compared to the train data. They can for example be rotated or shifted. This is also the case in the imagery being used in this research where buildings have different rotations and are not always located at the center of the used image (see Chapter 4). A technique to make the model less sensitive to these kinds of modifications, and thus prevent overfitting, is through data augmentation. Data augmentation generates synthetic data according to specified transformations, through which the number of data samples is increased. Data augmentation is only applied on the training set, not on the validation or test set. Common transformations are rotation, random cropping, noise insertion, and changing of brightness [61]. Several augmentation schemes are implemented in this research, as explained in Section 5.7.2.

3.2.7 Batch normalization

It is a common practice to rescale the input data to have a zero mean and unit variance. However, as the data progresses through the network, the distribution of the input to a layer changes, which is termed the *internal covariance shift*. This causes the network to learn slower and degrades the accuracy. A method named *batch normalization* was proposed by Ioffe & Szegedy [62], which reduces the internal covariance shift. Batch normalization changes the output of an activation function to have zero mean and unit variance before feeding it to the next layer. Batch normalization has several advantages and it is a common practice to use in CNNs. The advantages are that it enables a larger learning rate, makes the network less sensitive to parameter initialization, acts as a regularizer, makes it possible to use saturating nonlinear functions as activation functions, and can speed up the training process [61].

3.2.8 Model architectures

By combining all the previously discussed building blocks, a model architecture can be constructed. This model architecture defines the hypothesis space, though in the case of DL this space is very complex and cannot be captured in a formula. The best performing model architectures have become more complex and creative over the last years. A good indication for the best performing CNNs is by looking at the well-performing entries of the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [63]. Imagenet is a large dataset that became the worlds' benchmark for image classification and ILSVRC is a competition that ran from 2010 to 2015 on this dataset and saw many big advances in deep learning (DL).

In 2014 the architecture named VGG [64] performed well in this challenge. VGG is a relatively simple architecture where convolutional layers are stacked and the filter size increases over the course of the network. While relatively simple compared to its successors, it often performs well on a wide range of problems. Different versions of VGG exist, of which the most common is VGG-16, which indicates that it consists of 16 layers. The main disadvantage of VGG is the huge computational cost, in terms of time as well as memory. Another popular model is the Inception model, which was developed by Szegedy et al. [65] and was submitted to ILSVRC in 2015. This architecture uses "Inception blocks" which convolve the output of an activation function through several smaller filters instead of one deeper one. This lowers the computational costs and is better at handling objects of multiple scales. Moreover, the usage of 1×1 filters also provides regularization. Several improvements have been made to the Inception model, and the most popular version is the Inception v3 model, which is schematically depicted in Figure 3.7. The inception model is a component of the architecture used in this research (Section 5.3).

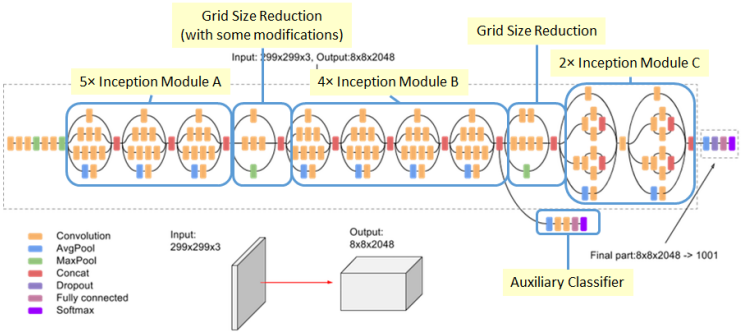


Figure 3.7: Architecture of the Inception v3 model, from [66]

3.3 TRANSFER LEARNING

The performance of deep learning models has improved significantly over the last years, where they have shown supreme performance in many fields. Those supreme models are in general trained and tested on the same domain and task. When the data suddenly looks different during test time, models often fail, i.e. a big challenge of present deep learning is the *generalizability*. To work well on different data, new, labeled data has to be gathered similar to the test data and the model has to be retrained with that data. This is troublesome since gathering labeled data is often a laborious task. A subfield that tries to overcome this problem of not being able to generalize is *transfer learning*, which works with the idea that the learned knowledge can be transferred to a new domain or task, as illustrated in Figure 3.8.

The most common definition of transfer learning is defined by Pan and Yang [68]. The data can be split to *source* and *target*, where the source is the original data and

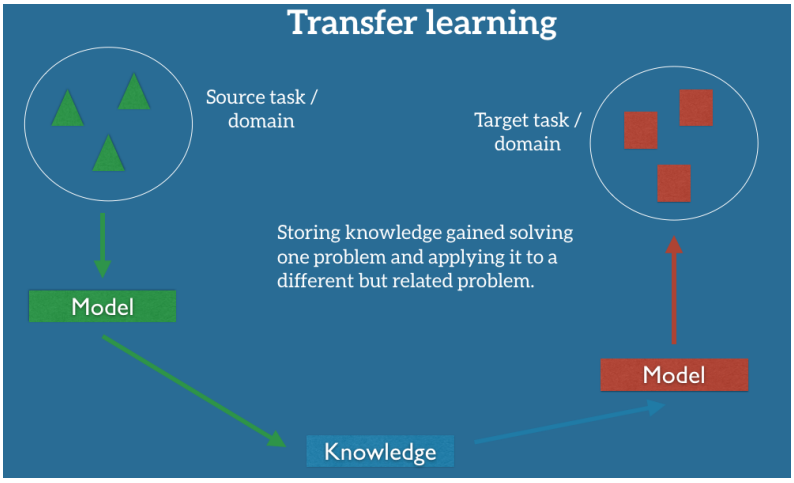


Figure 3.8: Illustration of the concept of transfer learning. Adapted from [67].

target the new test data. Both consist of a domain $\mathcal{D}$ and a task $\mathcal{T}$. The domain $\mathcal{D}$ consists of the feature space $\mathcal{X}$ and a marginal probability distribution $P(X)$. X denotes the input data, where $X = \{x_i, \cdots, x_n\} \in \mathcal{X}$. The task $\mathcal{T}$ consists of a label space $\mathcal{Y}$ and a conditional probability distribution $P(Y|X)$. $P(Y|X)$ is learned during training by using training pairs $x_i \in X, y_i \in Y$. Given a source domain $\mathcal{D}_s$, a target domain $\mathcal{D}_t$, a source task $\mathcal{T}_s$ and a target task $\mathcal{T}_t$, the goal of transfer learning is to learn the conditional probability distribution of the target, $P(Y_t|X_t)$ using the knowledge gained from $\mathcal{D}_s$ and $\mathcal{T}_s$. While transfer learning can bring great improvements in performance, it should not be applied blindly as it has been shown empirically that transfer learning can hurt performance when the target and source domain are too unrelated [69].

A division in transfer learning can be made in if $\mathcal{T}_s \neq \mathcal{T}_t$ or $\mathcal{D}_s \neq \mathcal{D}_t$. Where the first is named *inductive* transfer learning and the latter *transductive* transfer learning. If $\mathcal{T}_s \neq \mathcal{T}_t$ it means that $\mathcal{Y}_s \neq \mathcal{Y}_t$ or $P(Y_s|X_s) \neq P(Y_t|X_t)$. For both cases, labeled data of the target task is needed, the main goal is to limit this data. $\mathcal{Y}_s \neq \mathcal{Y}_t$ is often applied by using pre-trained weights from another but related problem, such that the model will converge faster on the target task. A classic example for image classification is to use weights that are pre-trained on the ImageNet dataset [63]. The motivation being that those weights are already able to recognize features that are important for the classification of images, such as edges. It has been shown many times that using those pre-trained weights indeed leads to better performance while not overfitting [70]. This technique is also applied in this research, as further described in Section 5.3. $P(Y_s|X_s) \neq P(Y_t|X_t)$ indicates that the class distribution is not the same between the source and the target, which is commonly the case. Examples to account for this are over- and undersampling, but this case is often not the main focus of transfer learning research.

When $\mathcal{D}_s \neq \mathcal{D}_t$ either $\mathcal{X}_s \neq \mathcal{X}_t$ or $P_s(X) \neq P_t(X)$. In this case, it is often assumed that no labeled data of the target domain is available, but it is in general assumed that (part of) the unlabeled target data is available during training [68]. When $\mathcal{X}_s \neq \mathcal{X}_t$ the input space is not the same, in natural language processing (NLP) this can, for example, occur when the source and target are in different languages, in computer vision the task could be to predict the number of objects in the image, but in the source, this could be cars while in the target bicycles. When $P_s(X) \neq P_t(X)$, the marginal distributions are different and this is called domain adaptation [71]. An NLP example is if the domains have different topics, in computer vision it can, for example, be images with bicycles taken inside vs outside. However, the line of when $\mathcal{X}_s \neq \mathcal{X}_t$ and when $P_s(X) \neq P_t(X)$ is rather ambiguous. This division is also unclear in the context of this research, but it is assumed that the domain of each disaster is different, whether it is because the marginal distributions are the

input space are not the same. The magnitude of this difference depends on the disasters, for example if they show the same type of damage. A set of disasters, i.e. different source domains, is used to train the model where after the model is tested on another target domain, i.e. the test disaster. The experiments done with this set-up of transfer learning are further described in Section 6.2.

3.4 IMBALANCE

In many real-world situations, the data is not balanced, meaning that not every label class contains the same number of samples. For example, in this research the largest fraction of buildings do not show damage. For the performance of ML models, this can cause a problem [72]. They tend to incline towards focusing on the majority class since this will in general lead to the smallest total loss.

In most applications this is not desirable behavior, e.g. this research often puts more value on correctly predicting the minority classes than the majority class. To partially overcome this problem, several techniques have been invented. While these techniques help, overcoming degrading performance due to imbalance is still an active field of research [73, 74]. The developed techniques can be divided into two categories: *data level* methods and *classifier level* methods [72]. The data level methods alter the training dataset with the aim that the model will find the desired minimum. The classifier level methods leave the dataset intact, but change the algorithms. Below one method of each category is elaborated upon, *resampling* as data level method, and *cost-sensitive learning* as a classifier level method. Both techniques are also implemented in this research (Section 5.7.1).

3.4.1 Resampling

With *resampling*, the data points are sampled such that a certain distribution between the different classes is established. There are two methods to achieve this, by *over- and undersampling*. By oversampling, data points from the minority class(es) are duplicated until the desired distribution is reached. With undersampling, a part of the data points of the majority class(es) are removed until the distribution is reached. The most common method to choose which data points to add or remove is by random selection [72]. However, there are also more sophisticated methods such as focusing on samples that are close to the neighboring classes. Another well-known method for oversampling is SMOTE [75], which creates augmented samples of the minority class instead of solely duplicating them.

3.4.2 Cost-sensitive learning

The core of *cost-sensitive learning* is that the loss function can be multiplied by a weight, where the weight depends on the predicted and/or true label [76]. In the most elaborate version, a separate weight can be given for each combination of true label and predicted label, which gives the different (mis)classifications different weights. This enables to take prior context information into account, but it can also be used to tackle the imbalance problem. The best way to implement cost-sensitive learning in DL is still an active area of research [77].

3.5 PERFORMANCE MEASURES

ML models are typically evaluated in three ways, by the loss function (see Sections 3.2.4 and 3.4.2), the performance measure on the validation set, and the performance measure on the test set. This section discusses several choices of performance mea-

sures for the validation and test set. A different performance measure can give a very different impression of a model and thus, the used measure has to be chosen such that it is in line with the purpose of the model.

3.5.1 Accuracy

Accuracy is the most simple measure which is used often, but can result in a too optimistic picture of the predictive value of the model. This is especially the case when dealing with imbalanced classes. An accuracy of 80% might sound good, but if the dataset consists of 80% of one class then always predicting that class will already result in an accuracy of 80%, while the model does not have any predictive value in that case. Nevertheless, it is still a very commonly reported measure, as we also saw in previous research on automated damage assessment (Section 2.7). Because the data used in this research is highly imbalanced, solely judging by accuracy is not suited. Hence, different measures are also considered as explained in the next sections. Section 5.6 explains in more detail which performance measures are used and why.

3.5.2 Confusion matrix

To get a detailed picture of how the data points are classified, a *confusion matrix* is a good tool to use since it shows for each combination of the true and predicted label how many data points are classified to that combination. Figure 3.9 shows an example of such a matrix of a binary classification problem. This matrix shows the predicted label on the x-axis and the true label on the y-axis. The four squares can be divided into *true positives (TP)*, *true negatives (TN)*, *false positives (FP)* and *false negatives (FN)*. A *positive* is a data point classified as class *one* and a *negative* data point classified as class *zero*. *True* indicates that the data point is correctly classified, while *false* that it is incorrectly classified. In a binary confusion matrix, the diagonal contains the TP and TN, and the other two cells the FP and FN. The confusion matrix can easily be extended to a multiclass problem. In this case, for each class the diagonal value of the class of interest indicates the TP, the sum of the other values in the row the FN, the sum of the other values in the column the FP, and the sum of values in other columns and rows the TN.

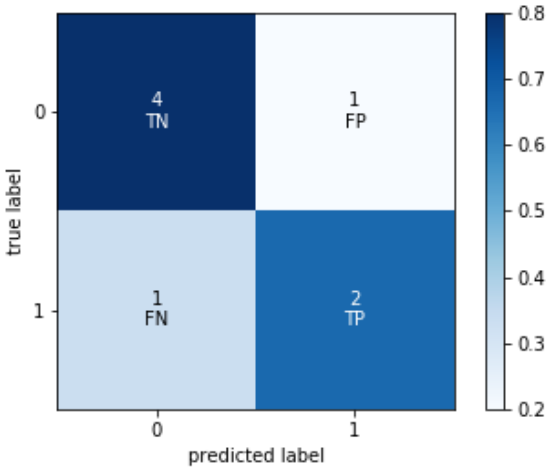


Figure 3.9: Example of a confusion matrix.

3.5.3 Precision, recall and F1-score

A confusion matrix is a good tool to understand the predictive value of the model better, but the large amount of numbers makes it challenging to compare results between different experiments. Therefore, measures have been invented that aim to summarize a confusion matrix in a few numbers. Note however that with this summarization, one will lose information.

Three of the most common measures are *precision, recall, and F1-score*. Precision is calculated as the true positives over the sum of all positives, this answers the question "What fraction of the positive predicted samples is actually positive?". Recall is defined as the true positives divided by the actual positives, and this answers the question "Which fraction of the positives are correctly captured by the model?" Mathematically, these can be expressed as:

$$\text{Precision} = \frac{\text{true positive}}{\text{true positive + false positive}} = \frac{\text{true positive}}{\text{predicted positive}}\,, \tag{3.12}$$

$$\text{Recall} = \frac{\text{true positive}}{\text{true positive + false negative}} = \frac{\text{true positive}}{\text{actual positive}}\,. \tag{3.13}$$

The F1-score combines the precision and recall as a harmonic mean:

$$\text{F1-score} = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision + Recall}}\,. \tag{3.14}$$

Which set of measures is the most suitable, is highly dependent on the purpose of the model. For example, one wants to make sure that no instances of a certain class are missed, one wants to focus on getting a high recall for that class. For this study, this can apply for certain purposes, for example, if it is important that everyone who has damage to their house gets certain aid. Simultaneously if the overall performance is more important, the F1-score often gives a good insight.

When expanding to multiclass classification, the measures should be averaged over all classes to get one score. There are different methods to take the average, of which the most commonly used ones are the *micro average, macro average, weighted average*, and *harmonic mean*. The micro average is computed by first summing all occurrences of the value of interest, e.g. TP, over all classes and then calculating the total score of interest, e.g. recall. The other three methods use a one-vs-all approach. This means that if the measure, e.g. recall, of one class is calculated, the problem is binarized such that the class of interest counts as the positive class whereas all other classes are taken together as the negative class. This leads to a number for the measure per class, after which these numbers are averaged. This is done differently for each of the three averaging methods:

$$\text{Macro average} = \frac{\sum_{i=1}^{C} S_i}{C}\,, \tag{3.15}$$

$$\text{Weighted average} = \frac{\sum_{i=1}^{C} N_i \cdot S_i}{\sum_{i=1}^{C} N_i}\,, \tag{3.16}$$

$$\text{Harmonic mean} = \frac{C}{\sum_{i=1}^{C} (S_i + \epsilon)^{-1}}\,, \tag{3.17}$$

where S is the value of interest, e.g. precision, C the number of classes, N_i the number of data points per class, and ϵ a very small number to prevent division by zero. The macro average is most commonly used. The weighted average weights the class measures by the number of occurrences and thus gives more weight to the majority class. The characteristic of the harmonic mean is that it drops drastically if the score for one of the classes is low. Again, it depends heavily on the purpose of the prediction, which method is the most suitable. For example, if the model must perform well across all classes, the harmonic mean is a good measure.

3.5.4 ROC curve and AUC score

receiver operating characteristic (ROC) curves are another technique to measure the performance of a classifier, where the focus is on the ability of the classifier to separate the classes. A ROC curve makes use of the predicted probabilities that a data point belongs to each class and is mainly applicable in a binary setting. Whereas for accuracy, the class with the highest probability is always chosen as output label, a ROC curve differs this classification threshold and shows the true positive and false positive rate for a range of thresholds. See Figure 3.10 for an example ROC curve. If the line would follow the dashed black line, it would indicate a model that gives random predictions. By testing the performance for several classification thresholds, a more general impression of the ability of the model to separate the classes is shaped. This is useful because, assuming a binary setting, the optimal classification threshold might be different from 0.5. This way of measuring performance is especially useful when working with imbalanced data, since in those cases the optimal threshold is often shifted and thus the ROC curve is less sensitive than accuracy to imbalance.

Since comparing ROC curves across experiment settings is hard, the ROC Curve can also be captured in one number, the *area under the curve (AUC)*. This is literally the area under the ROC curve. The optimal model has a AUC of 1, whereas a AUC of 0.5 indicates random performance. The biggest advantage of the AUC is its decreased sensitivity to an imbalance in the data, and thus is a preferred way to measure the performance of a binary classifier compared to accuracy [78]. Unfortunately, ROC curves are not very well suited for multiclass classification, since there is not a consistent threshold per class that always assigns a data point to that class, the predicted label is simply the class with the highest probability. This results in the behavior that changing one threshold can influence the classification in more complex ways than with binary classification. Hence, the ROC curve and the AUC lose their interpretability.

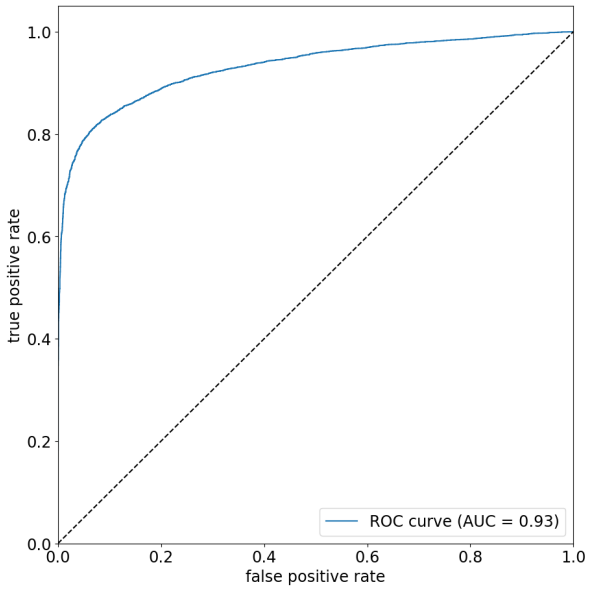


Figure 3.10: Example of a ROC curve. The dashed line indicates random performance.

3.5.5 Qualitative analysis

Complex ML models can perform in a certain way for many reasons. One can try to capture this in a number, but the number does not always tell everything. Therefore often the best way to get a feeling for the model is to simply inspect a sample of

the outcomes and compare apparent differences between correctly and incorrectly classified data points. Thus, this will be one of the methods used in this research. The main tools being used in this research to guide a meaningful qualitative analysis are QGIS[1] and Google Facets[2].

Chapter summary

- A machine learning (ML) problem can be defined by its feature space, label space, hypothesis space, and loss function
- A deep learning (DL) network consists of layers, activation functions, a loss function, and an optimization function
- Overfitting is a common problem and can be circumvented by several regularization techniques
- Imbalance in data can cause difficulty for the model and several techniques exist to limit this difficulty
- The goal of transfer learning is to use the knowledge of the source domain and task to let the model perform well on the target task, for which limited information is known
- The choice of performance measure highly depends on the purpose of a model and a range of methods exist

1 https://qgis.org/
2 https://pair-code.github.io/facets/

4 DATA

A famous saying in ML states: "Your model only gets as good as its data". This indicates that it is very important to understand your data. In this research the xBD dataset is used (see Section 2.6). This chapter briefly describes the few preprocessing steps of the data that have been taken in this research. Next, it dives into the data to discover several quantitative and qualitative characteristics.

4.1 PREPROCESSING

In total the released xBD dataset contains 19 disasters, adding up to 316,114 unique buildings. For this research, we have decided to leave the occurrences of fires and earthquakes out of the dataset. This is because these disasters result a very different type of damage and it is not 510's priority to produce a damage assessment on these at the moment.

We also leave the unclassified buildings out of the dataset, since this is not the main focus of what the model should learn. These are buildings that could not be well-marked by human annotators. For example, buildings obscured by clouds and buildings that appear more finished after than before the disaster [33]. Lastly, we remove the buildings that have more than 10% zero-valued, i.e. black, pixels with the reasoning that from this data not much information can be retrieved. As it turns out, about 20% of the data is disregarded. This way of selecting the data is debatable, and different techniques can be experimented with in the future.

4.2 DATA ANALYSIS

The disasters included in this research and some of their characteristics can be seen in Table 4.1. After the selection of data, we are left with 13 disasters, totaling 175,000 buildings. These disasters are spread over three regions, North America, Central America, and Asia, and can be divided into four damage types, *flooding*, *wind*, *tsunami*, and *volcano*. In the xBD dataset, each disaster only belongs to one damage type, where a hurricane only cause wind or flooding damage. The buildings are labeled according to the Joint Damage Scale (Section 2.3) and contain four classes: *no damage*, *major damage*, *minor damage*, and *destroyed*. When looking at the table, we can see a large difference in the number of buildings per disaster. This means that for some disasters much more training data is available than for others. Moreover, the class distribution differs considerably per disaster, while on average most buildings belong to the *no damage* category.

Figure 4.1 shows the before imagery, after imagery, and labels of a small area impacted by the Joplin tornado and Nepal flooding. It can be seen that both areas are heavily damaged, while the damage looks completely different. In addition, we see a clear difference in the distribution of damage and the spatial ordering of the buildings. We can examine the appearance of the buildings and damage further by inspecting individual buildings. Figure 4.2 shows an example of a building in each damage class after the disaster for the Joplin tornado and Nepal flooding. For the Joplin tornado, there is a clear visual trend, as the damage level increases the damaged buildings appear browner, indicating broken roofs. In the *minor damage*

Region	Disaster	Damage type	Number of buildings	Class distribution
Asia				
	Nepal flooding	flooding	29808	75/13/11/1
	Palu tsunami	tsunami	24119	83/0/2/15
	Sunda tsunami	tsunami	11682	98/0/1/1
Central America				
	Guatemala volcano	volcano	493	94/1/1/4
	Hurricane Matthew	wind	9506	19/52/16/12
North America				
	Moore tornado	wind	18858	87/4/2/7
	Tuscaloosa tornado	wind	12579	75/14/3/8
	Joplin tornado	wind	12165	56/15/7/22
	Hurricane Michael	wind	20046	64/24/9/3
	Hurricane Florence	flooding	5243	77/2/20/1
	Hurricane Harvey	flooding	21516	50/12/36/2
	Midwest flooding	flooding	7161	96/2/1/1
	Lower-Puna volcano	volcano	2113	81/2/1/16
3	**13**	**4**	**175289**	**72/12/10/6**

Table 4.1: Details of the disasters included in this research. The column *damage distribution* shows the percentage of samples belonging to the classes *no damage*, *minor damage*, *major damage* and *destroyed*, respectively.

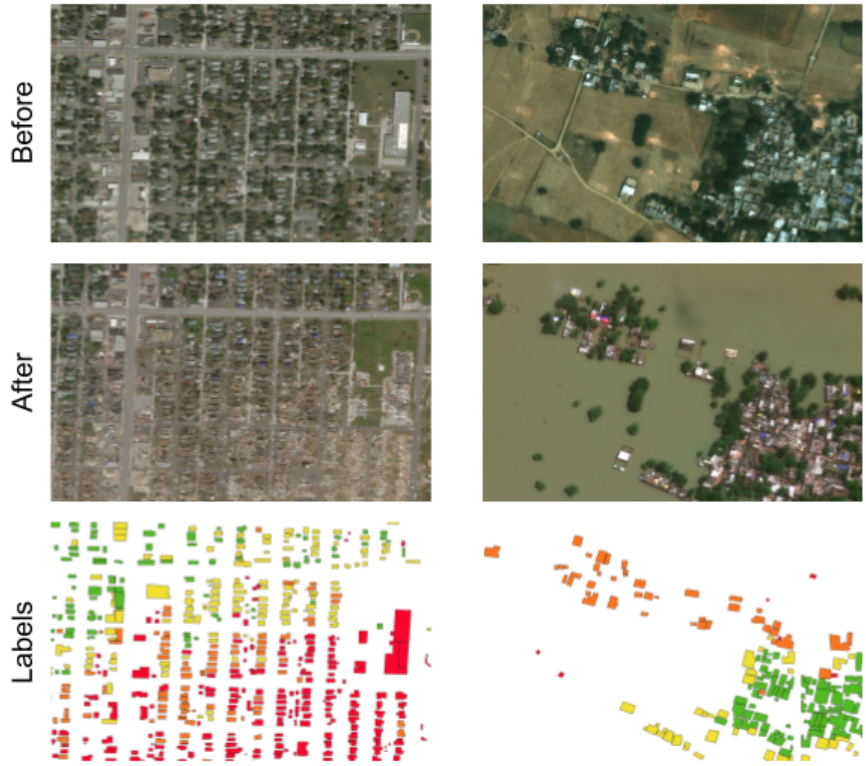


Figure 4.1: Examples of damage caused by the Joplin tornado (left) and the Nepal flooding (right). They show wind and flood damage, respectively. The color in the bottom row show the damage labels where green indicates *no damage*, yellow *minor damage*, orange *major damage*, and red *destroyed*.

class, a characteristic sign is the blue tarps on the roof. For the Nepal tornado, the color distribution is different, as the damage class increases, the images appear more green, which indicates the presence of water and thus (partly) flooded buildings. Moreover, it can be seen here that the images seem a bit less sharp and the buildings are not always centred. This is due to the relatively high off-nadir angle, and the often smaller building sizes compared to the Joplin tornado. While these examples show a good indication, it is important to bear in mind that the looks of damage and buildings are rather diverse per class, especially for the Nepal flooding.

Figure 4.2: Examples of the after image of a building for each of the four damage classes for the Joplin tornado (left) and Nepal flooding (right).

One instance of different looks, is the clarity of the image for the human eye. Figure 4.3 shows two buildings after the Joplin tornado that belong to the damage class *destroyed*, but the clarity of the image for the human eye is very different due to the difference in the sizes of the buildings.

Figure 4.3: Examples of two destroyed buildings in the Joplin tornado. On the left, the damage can clearly seen by the eye, the right is very blurry.

Apart from the small building sizes, there are other characteristics of some data points that are not optimal and which the model has to learn to manage. The building outlines are labeled on the pre imagery. The post image can be shifted, mainly due to differences in the off-nadir angle of the satellite. This can cause the polygon not to overlap with the actual building and thus, in this case, only partial information is given to the model. Another challenge is mislabeled buildings. The manual labeling of buildings, in this case done by humans from aerial imagery, will always remain a very subjective task. Whether a building belongs to one or another class is debatable, but clear mistakes in the labels can be seen for some buildings where the building is split into two polygons and those polygons have been assigned different labels. Though this is an exception, it indicates that the model also has to deal with this kind of ambiguities in the data. Lastly, the differences in satellite parameters, as explained in Section 2.4, can make images look very different. The model has to learn to handle these differences in resolutions, angles, and elevations. Additionally, the season and time delay can influence the looks of the image, and clouds covering buildings are another problematic factor. Some examples of these challenges are shown in Figure 4.4.

Table 4.2 shows an overview of the range of parameters across all data. One thing that can be observed from this table is that the duration of disasters differ,

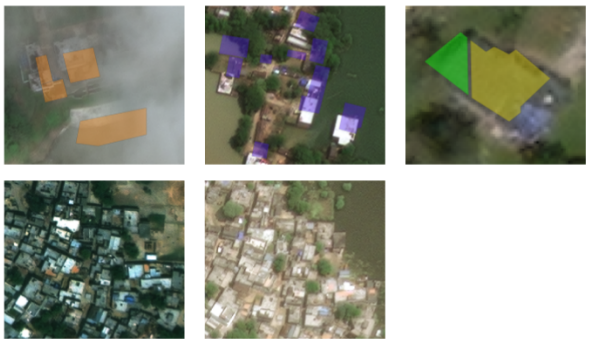


Figure 4.4: Examples of challenges in the data. Top: clouds, misaligned polygons, mislabeled building. Bottom: difference in illumination between pre and post image

some take place during one day, namely tsunamis and tornadoes, while others can take place over a long period, especially floods. This duration also depends on whether there are consecutive events, such as a landslide after a hurricane. Because of differences in the duration of disasters, the time delay between the beginning of the disaster and the retrieval of the post satellite imagery sometimes results in a skewed conclusion. Therefore, when exactly the post imagery is needed depends on the disaster type and the purpose, as we saw in Section 2.2. However, the minimum time-delay in this dataset is six days, and thus nowhere close to the theoretical one to three days resolution that Maxar mentions as its temporal resolution of the satellites (see Section 2.5). This longer time-delay does not impact this research, but it does impact the practical implementation and thus should be investigated further.

Parameter	Median	Mean	Standard deviation
Image pairs per disaster	2.00	2.38	1.50
Disaster duration (days)	16.00	51.29	56.49
Panchromatic resolution (m)	0.51	0.52	0.12
Off-nadir angle (degrees)	21.75	22.92	10.28
Sun elevation (degrees)	62.53	60.82	11.04
Target azimuth (degrees)	177.35	173.43	106.68
Sun azimuth (degrees)	140.52	132.22	31.69
Building footprint (m^2)	207.90	297.63	752.33
Delay pre and post imagery (days)	351.00	561.71	515.37
Delay start disaster and post imagery (days)	12.00	31.6	48.69

Table 4.2: The median, mean and standard deviation of several important parameters across the set of 13 disasters used.

Chapter summary

- The preprocessed dataset consists of 13 disasters, four damage types, and more than 175,000 buildings
- There is a large class imbalance in the data
- There are clear differences in damage characteristics across disasters types
- While of good quality in general, there are also challenging aspects of the data that the model has to learn to deal with

5 METHODS

Without a model, no predictions can be made. The model used and further developed during this research is a model that has been initiated by 510. This chapter introduces the previous development of the model, the architecture, and its hyperparameter settings. Next, the machine learning (ML) problem is formalized, performance measures are chosen, and modifications that were made to the model described.

5.1 PREVIOUS DEVELOPMENT OF THE MODEL

After the Irma hurricane struck on St. Maarten in 2017, 510 started developing an automatic damage assessment model, for which the main development was during D. Kersbergen's thesis [11]. He used airborne and UAV optical imagery, and satellite SAR imagery, all collected on St. Maarten before and after hurricane Irma. He implemented two models: a model based on the remote sensing technique of coherence change detection, adapted from [44], and a convolutional neural network (CNN) , adapted from [79]. With his settings and data, he showed that the remote sensing technique was able to learn to classify damage on the data of St. Maarten up to a certain extent. His CNN did not learn to recognize damage for a not completely understood reason.

Later an improved CNN model was developed during a hackathon. Volunteers of 510 have since then been busy improving this model, and have shown that it can learn to detect damage related features on the St. Maarten optical data. The code is open source and can be found at https://github.com/rodekruis/caladrius. This model serves as a baseline for this research.

5.2 INPUT AND OUTPUT DATA

The model takes as input the pixels of a building before and after the disaster, where the image consists of the red, green, and blue band. The individual building images are generated by taking a bounding box around the building polygon in the original imagery and then adding 20% to the width and height of the bounding box. Taking this extra 20% around the building is done because the surrounding area often contains important information about the damage, such as debris, and because the building polygons do not always precisely overlap with the buildings in the imagery. All cropped buildings are then re-scaled to have a width and height of 299 pixels since this is the input size of the model.

The model was built as a regression model, but for this research, the option for classification has been added. This research solely focuses on classification, and thus regression is not further discussed. For the classification, it means the output size of the model equals the number of classes that should be predicted. In this research, the number of classes is either two or four. For each class, the model outputs a probability of the building belonging to that class and a predicted label, which equals the class with the highest probability.

5.3 MODEL ARCHITECTURE

The model consists of two convolutional neural network (CNN) networks, followed by a block of fully connected layers. A schematic overview of the architecture is shown in Figure 5.1. One CNN network takes the pixels of the building in the *before* image as input, while the other takes the pixels in the *after* image. Both CNNs output a feature vector of size 512. The two feature vectors are then concatenated, after which two blocks of fully connected layers with the ReLU activation function, batch normalization, and dropout follow. The last layer is yet another fully connected layer, followed by a Softmax activation to produce the output. The motivation of this architecture is that the CNNs learn to extract the typical features of a building before and after the disaster. The following layers then learn which precise features from each CNN and which differences between the features of the two CNNs indicate the different damage classes.

The architecture of the CNN networks is the Inception V3 network (see Section 3.2.8), where the original output size of 2,048 features is replaced by 512. The weights of the Inception models in this research are pre-trained on ImageNet, after which they are re-trained on our data during the training process. This is a common transfer learning practice in training deep models for computer vision since it normally results in faster convergence (see Section 3.3).

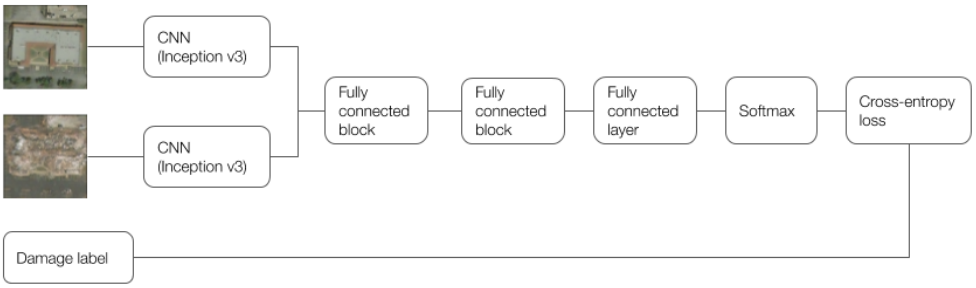


Figure 5.1: Architecture of the used model. The fully connected blocks consist of a fully connected layer with the ReLU activation function, followed by batch normalization, and dropout.

5.4 EXPERIMENTAL DETAILS

Besides the architecture, several other choices related to the model have been made, as described in Chapter 3. As the loss function, the cross-entropy loss is chosen. The optimization method being used is Adam, using the suggested hyperparameter settings of the authors: $\beta = 0.9, \rho = 0.999, \epsilon = 1 \times 10^{-8}$. The final set of weights is chosen by selecting that combination of weights that gains the highest macro F1 on the validation set across all epochs. The other hyperparameters of the model are shown in Table 5.1. These hyperparameters have been selected before this research and these were then set to produce the best results. All hyperparameters are kept constant throughout the experiments unless mentioned differently. All the code was written in Python, and as a deep learning framework PyTorch was used. The experiments were run on a single GPU, most of the experiments using the Nvidia V100 Tesla card with 32GB memory.

Epochs	100
Batch size	32
Dropout probability	0.5
Learning rate	0.001
Learning rate decay	0.1 × learning rate if validation loss does not change for 10 epochs

Table 5.1: Hyperparameters of the model.

5.5 DEFINING THE MACHINE LEARNING PROBLEM

Now that all the details of the model and the data have been defined, the ML problem at hand which consists of the feature space, label space, hypothesis space, and loss function (see Section 3.1) can be formally described.

The input consists of the pixels of a pair of pre and post imagery containing a building. Each building image has a width and height of 299 pixels and a depth of three, consisting of the red, green, and blue bands. This results in the feature space $\mathcal{X} = \mathbb{R}^{2\times 299^2\times 3}$. The label space consists of the damage labels, thus for the multiclass case $\mathcal{Y} = \{\textit{no damage, minor damage, major damage, destroyed}\}$ and for the binary case $\mathcal{Y} = \{\textit{not destroyed, destroyed}\}$. The hypothesis space is all the mappings that can be made by the defined architecture, which is a very complex space and cannot be put to a formula. Lastly, the loss function is the cross-entropy loss.

5.6 PERFORMANCE MEASURES

Performance measures are used to assess the performance of the model. There are different methods for measuring the performance (see Section 3.5) and it is important that the most appropriate measures are chosen. In this research, accuracy, AUC, and recall of the destroyed class were chosen as measures for the binary case. Accuracy was chosen because it is the most commonly reported performance measure in other studies. Simultaneously, it is not informative for our situation due to the high imbalance of classes. Hence, AUC is also chosen as a measure since it is less sensitive to imbalance. Lastly, the recall over the destroyed class is important, since for several purposes of a damage assessment it is especially important that the destroyed class is detected well.

For the multiclass situation, macro F1, harmonic F1, and the recall over all the classes are used. Macro F1 was chosen because it is the most common measure for the multiclass setting and gives a good idea of the overall performance. The harmonic F1 gives more importance to equal performance across all classes, which is relevant in our context if we want to make sure none of the classes is very wrongly classified. Moreover, the recall per class is looked at, because to assess the suitability of the model for practical purposes it is important to know to what extent the model is recognizing the damage in each class. Besides these numeric measures, more analysis for both the binary and the multiclass case is performed by using confusion matrices, distribution plots, and visual inspection of individual images.

5.7 MODIFICATIONS TO THE MODEL

Several modifications were made to the original model. Features were added to experiment if they would improve the model. In addition, some features were removed to research if they have added value to the model. The modifications can

be divided into three categories: *imbalance*, *augmentation*, and *architecture*. These are further described below and the results of the experiments with the modifications can be found in Section 6.3. These modifications are only implemented in the experiments in Section 6.3 and are thus not part of the model the experiments in Sections 6.1 and 6.1.4.

5.7.1 Imbalanced data

As shown in Chapter 4, the data is highly imbalanced meaning that the number of samples in each class is not equal. This is a common but big problem in ML since often the model will start to overfit on the majority class. In our case, this is especially troublesome since the majority class in most cases is *no damage*, whereas for most user needs it is especially important that data points belonging to the class(es) that show damage are correctly predicted. Section 3.4 elaborated upon two approaches to tackle the imbalance problem: resampling the training data and cost-sensitive learning. Both approaches are implemented in this research.

5.7.1.1 *Resampling*

For this research, balanced resampling was applied. This means that a combination of up- and downsampling was done such that all classes contained the same amount of samples. In this implementation, the total number of samples was left unchanged in the resampled data compared to the original data. This means that in the multiclass setting, since there were four classes, after resampling each class contained 25% of the samples, where the magnitude of this 25% amounted to one fourth of the size of the original data set. The up- and downsampled data points were chosen randomly and no augmentation was applied.

5.7.1.2 *Cost-sensitive learning*

With cost-sensitive learning a different weight can be assigned to each combination of the ground-truth and predicted labels (see Section 3.4.2). Here we solely focus on accounting for the imbalance and therefore assign the weight solely based on the ground-truth label and not on the predicted label. The approach implemented in this research uses the inverse median frequency. Let f_c be the class frequency of class c and f_m be the median of all class frequencies. Then the inverse median frequency for class c, w_c, can be defined as

$$w_c = \frac{f_m}{f_c} \,. \tag{5.1}$$

The resulting weighted loss of data point n is then defined as

$$\tilde{L}_n = w_c \cdot L_n \,, \tag{5.2}$$

where c is the class in which the data point n belongs.

5.7.2 Data augmentation

Buildings in satellite imagery can appear different within and between images due to how the building, Sun, and satellite are located. For example, the buildings can be shifted or rotated relative to other buildings. To make the model less sensitive to these differences, data augmentation can be applied (see Section 3.2.6.3).

The original version of the model already applies an augmentation scheme, as shown in Table 5.2. The augmentation scheme is the same for the pre and post disaster image and was implemented in PyTorch, which computes those augmentations on the fly for each batch. To test if the augmentations improve performance, a

Original augmentation scheme	Scheme adapted from [50]
crop [8-100% of original size]	rotation [-40°,40°]
horizontal flip (p=0.5)	translation $[-0.2, 0.2] \times$ image size
vertical flip (p=0.5)	shear [-11.5°,11.5°]
rotation [-90°,90°]	horizontal flip (p=0.5)
	crop [80-100% of original size]

Table 5.2: The augmentation schemes of the original model in this research (left) and the one adapted from [50] (right). The transformations are shown in the order they are applied.

model without any augmentations was tested. Moreover, the precise set of augmentations as used in [50] was implemented, since they showed that it improved their performance, and thus it is interesting to test if this also applies to the model used in this research. The augmentation scheme of [50] is similar to the original version of the model used in this research, but with small differences, as shown in Table 5.2.

5.7.3 Model architecture

For the model architecture it was chosen to use both pre and post imagery and to have separate weights for the two CNN networks. The reasoning for both choices was that the model could then extract as much information as possible. However, this leads to additional data requirements and higher model complexity. Therefore, these design choices should lead to an increase in performance for them to have added value. To test this, two changes were made to the architecture. The first architecture variant shares the weights between the two CNN networks, while the second solely uses the post imagery and not the pre imagery, and thus only consist of one CNN which takes the post building as input. For both modifications, the model has to be retrained and cannot use the weights of the model trained with the original architecture since the importance of different weights might change.

Chapter summary

- The model used in this research takes the pre and post image of a building and outputs a damage label, where the damage label is either binary or multiclass
- The architecture consists of two CNNs after which those outputs are concatenated, and given as input to two fully connected blocks, before returning the predicted damage label
- Several modifications to the model are tested related to class imbalance, data augmentation and the architecture

6 EXPERIMENTS AND RESULTS

This chapter describes the experiments that have been carried out and their results. For the first set of experiments, a model was trained on each of the thirteen disasters separately, and thereafter tested on the same disaster it was trained on. With this set-up, it is explored how the model performs on different disasters, for multiclass as well as binary classification, and how each disasters' parameters influence this performance. The second set of experiments analyzed the performance of the model on a test disaster that was not included in the train data. Different sets of training data were compiled to research how they influence the ability to transfer to another disaster. Lastly, the third set of experiments implemented several modifications to the data and the model, related to imbalance, data augmentation, and the model architecture, to analyze if those modifications lead to a change in performance.

6.1 PERFORMANCE ON INDIVIDUAL DISASTERS

Firstly, the performance on every individual disaster is tested. This means that the model is trained on 80% of the data for each disaster and thereafter tested on 10% of it. This experiment has three goals. Firstly, to test the predictive value of the model on satellite imagery. Secondly, to see if the performance differs per disaster, and lastly, if such differences exist, to analyze if the disasters' parameters influence the performance.

6.1.1 Binary classification

The model is first trained and tested on binary labels, since this should, in theory, be simpler to learn compared to multiclass labels. We binarize the labels to *destroyed* and *not destroyed*, where *not destroyed* contains the buildings from the classes *no damage*, *minor damage*, and *major damage*. This binary division is made because it is useful for some response actions, such as search and rescue. It is also the discrimination made in all previous research discussed in Section 2.7.

The model was trained and tested separately on each of the 13 disasters used in this research. The results are displayed in Table 6.1, where the AUC, accuracy, and recall of the destroyed buildings are used as performance measures. Based on the AUC, it can be seen that there are clear differences in performance between the disasters. When examining the accuracy, it can be seen that it is high for all disasters, varying between 0.895 and 1, but due to the large class imbalance this does not prove the predictive value of the model. The accuracy on four disasters (Sunda, Florence, Nepal, Midwest) is lower than the fraction of data points belonging to the majority class. This means that a naive model would reach a higher accuracy if it annotated all buildings of those disasters as the majority class, and thus, based on the accuracy measure, the trained model does not hold predictive value for those disasters. Nevertheless, for the other nine disasters it does perform better than always predicting the majority class and thus holds predictive value.

Moreover, it can be the case that the model learns damage-related features, but is able to distinguish them at a different threshold than the threshold of 0.5, which is the used threshold for the accuracy. To analyze if this holds, AUC can be used because it indicates the ability to separate the classes, regardless of the threshold.

Disaster	AUC	Accuracy	Recall destroyed	% destroyed	# buildings
Florence	0.496	0.990	0.000	1.0	512
Sunda	0.694	0.994	0.000	0.6	1184
Harvey	0.853	0.987	0.107	1.4	2048
Michael	0.891	0.976	0.148	2.6	2048
Matthew	0.894	0.895	0.540	15.0	928
Nepal	0.905	0.990	0.105	0.6	2944
Puna	0.962	0.943	0.800	18.2	192
Palu	0.972	0.957	0.830	14.4	2336
Joplin	0.987	0.941	0.915	21.1	1280
Moore	0.990	0.986	0.871	6.5	1920
Tuscaloosa	0.990	0.971	0.904	5.8	1248
Midwest	0.997	0.997	0.000	0.3	704
Guatemala	1.000	1.000	1.000	6.2	32

Table 6.1: Results of binary classification per disaster, sorted by AUC. The % destroyed and # buildings refer to the numbers in the test set.

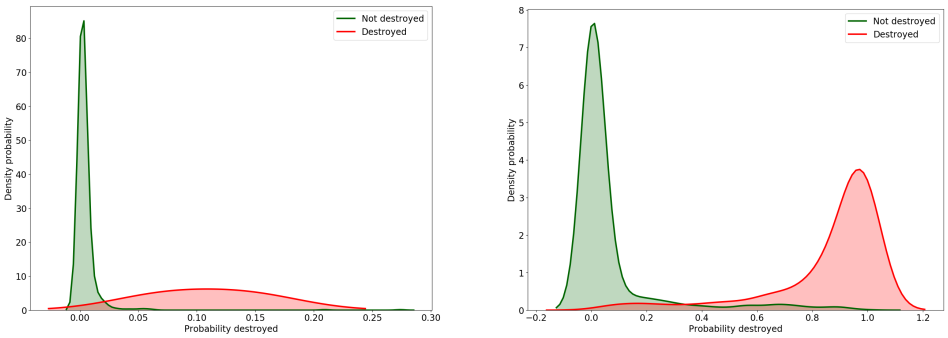


Figure 6.1: Distribution plots of the predicted probability of buildings belonging to the destroyed class for the Midwest flooding (left) and Joplin tornado (right). The red distribution is that of the data points with the *destroyed* ground-truth label, the green of the *not destroyed*. Pay attention to the different scales on the axes.

The fact that 12 out of 13 AUCs in Table 6.1 are larger than 0.5, shows that the model is able to separate the classes to some extent for all those 12 disasters. To understand what the optimal threshold is, we can look at the distribution plots over the *destroyed* and *not destroyed* class. Figure 6.1 shows this plot for the Midwest flooding and the Joplin tornado. For the Midwest flooding, it can be seen that the model can distinguish the two classes decently, but at an optimal threshold of around 0.025 instead of 0.5. Thus, for this specific disaster, the model would reach a higher accuracy and recall of the destroyed class if the classification threshold was to be changed from 0.5 to 0.025. When examining the Joplin tornado, the plot shows that the optimal threshold is closer to 0.5 which explains the relatively high accuracy and recall on the destroyed class. The high AUC for most disasters indicates that the model holds predictive value, but this value can only be used to its full potential if the optimal threshold for each disaster can be determined. The results show a general trend that a higher AUC and a higher recall of the destroyed class is reached when a larger percentage of buildings belong to the *destroyed* class, as illustrated in Figure 6.2.

6.1.2 Multiclass classification

Depending on the purpose of the damage assessment, a more precise granularity than binary separation is needed. While most papers on automated building damage assessment have solely focused on binary classification, arguing that it is too difficult to distinguish more categories from satellite imagery, this study investigates

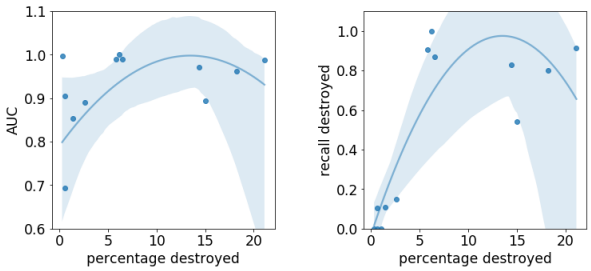


Figure 6.2: Scatter plots of the percentage of data points belonging to the *destroyed* class versus the AUC (left) and the recall on the *destroyed class* (right). Each dot represents one disaster, the blue line shows the best polynomial fit, and the blue area the 95% confidence interval.

the performance of the model on multiclass classification. A distinction is made between four classes, according to the original labels of the xBD dataset. These classes are *no damage*, *minor damage*, *major damage* and *destroyed*.

The set-up of the experiment is equivalent to the binary classification: a model is trained for each disaster and the split of training and test data is 80% and 10% respectively. To check if the model is learning, one can explore the training losses, see Figure 6.3. We observe that the loss decreases over time, which indicates learning. Moreover, a setting of 100 epochs seems reasonable since the loss stabilizes by then. To train for 100 epochs took between 20 minutes and 10 hours, depending on the training data size. Producing the test results took between three and 26 seconds.

The performance of the model on each disaster is displayed in Table 6.2. The harmonic F1, macro F1, and recall over each class are taken as performance measures. Similar to the binary case, one can observe large differences in performance between disasters across all measures. The most striking one is the harmonic F1 score, where five disasters have a score of zero. This stems from the fact that the harmonic F1 immediately goes to zero if the F1 score of one of the classes is zero. Looking at the table, for those five disasters, the recall of at least one of the classes is zero, which results in a zero F1 score for that class and thus a harmonic F1 of zero. The macro F1, the most common measure for multiclass classification, also shows large differences between the disasters. In the macro F1, a very low recall score in one class can be compensated by a high score in another class. This causes the ordering of disasters by the highest value of the performance measure to be different when judging on the macro F1 compared to the harmonic F1, which shows the importance in the choice of performance measure.

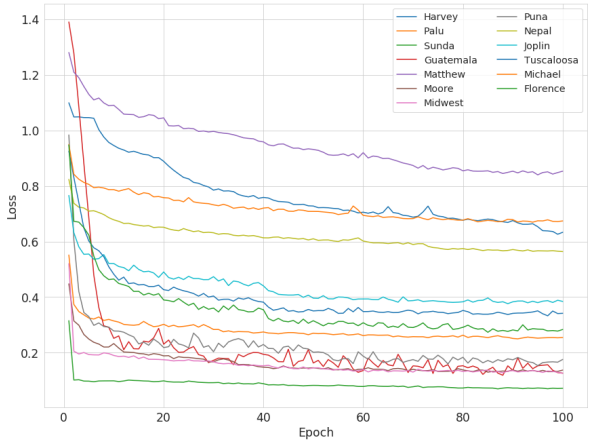


Figure 6.3: Plot of the training loss for each of the 13 disasters trained on multiclass labels.

	F1		Recall				Class percentage			
Disaster	Harm.	Macro	No	Min.	Maj.	Des.	No	Min.	Maj.	Des.
Palu	0.000	0.702	0.981	1.000	0.000	0.824	83.8	0.0	1.8	14.4
Sunda	0.000	0.332	0.997	1.000	0.000	0.286	98.7	0.0	0.7	0.6
Midwest	0.000	0.372	0.999	0.000	0.429	0.000	97.0	1.7	1.0	0.3
Puna	0.000	0.449	0.980	0.000	0.000	0.800	78.6	1.6	1.6	18.2
Florence	0.000	0.466	0.975	0.000	0.922	0.000	76.8	2.3	19.9	1.0
Nepal	0.073	0.482	0.989	0.010	0.459	0.368	74.3	13.0	12.1	0.6
Harvey	0.381	0.538	0.899	0.162	0.861	0.143	50.5	12.1	36.1	1.4
Michael	0.495	0.542	0.914	0.401	0.413	0.352	65.0	24.0	8.4	2.6
Matthew	0.543	0.577	0.382	0.858	0.315	0.676	16.9	50.0	18.1	15.0
Tuscaloosa	0.706	0.740	0.952	0.716	0.500	0.781	74.4	16.1	3.7	5.8
Joplin	0.758	0.787	0.961	0.705	0.516	0.907	55.9	15.6	7.4	21.1
Moore	0.771	0.802	0.995	0.500	0.766	0.887	87.1	4.0	2.4	6.5
Guatemala	1.000	1.000	1.000	1.000	1.000	1.000	93.8	0.0	0.0	6.2

Table 6.2: Results of multiclass classification per disaster, sorted by the harmonic F1 score. *No, Min., Maj., Des.* indicate the *no damage, minor damage, major damage, destroyed* classes respectively.

Looking closely at the recall per class, the majority class always has the highest score. Previous research has suggested that it is harder to distinguish the *minor* and *major damage* classes [32, 28], but the results of this experiment do not support that statement. Recall scores for those classes are not in all cases good, but not significantly worse compared to the scores of the *destroyed* class. It seems more likely that low recall scores for the *minor* and *major damage* classes can be attributed to the low percentage of data points belonging to those classes. This hypothesis is further explored in Section 6.1.4 where it is shown that this is indeed part of the explanation. From this experiment, it can thus be concluded that it is too simple to state that minor and major damage cannot be recognized well. This experiment shows that it highly depends on the disaster and the percentage of labeled samples whether this statement holds.

Due to the lack of a suitable performance measure for multiclass imbalanced data, it is hard to compare and understand the performance of different disasters by looking at the given performance measures. A confusion matrix is another method to understand performance. This matrix can especially be helpful in our case, where the classes are ordered and we thus rather have the model predict classes that are close to the actual label, e.g. it is preferred that the model predicts *major damage* than that it predicts *no damage* when the label is *destroyed*. Figure 6.4 shows the confusion matrices for two disasters, the Nepal flooding and Joplin tornado.

In the confusion matrix of Nepal, we can see that the model has a high tendency to predict *no damage*, even for *destroyed* samples. This is non-desirable behavior and gives the insight that the model is not learning the correct damage-related features. To understand where the mispredictions come from, a qualitative analysis can be done. Figure 6.5 shows four examples of misclassified buildings, two where the level of damage is overpredicted and two where it is underpredicted. From here we can see that some mispredictions are explainable. For example, the top left building does not seem to be fully surrounded by water due to the trees, judging from the area of the building crop. However, if zooming, it could be seen that the building is fully surrounded by water, leading to a categorization of *major damage*. From the given input to the model however, it is understandable that the building is labeled as *no damage*. In the bottom left, it is very hard to see the building due to the blurriness and the small building size. However, it is understandable that the building is predicted as *destroyed* instead of the real label *no damage* since the post image does appear completely flooded such that no building is visible anymore, and the white pixels on the pre image might be interpreted as the original building. The right bottom figure overpredicts damage, probably due to the lightning that

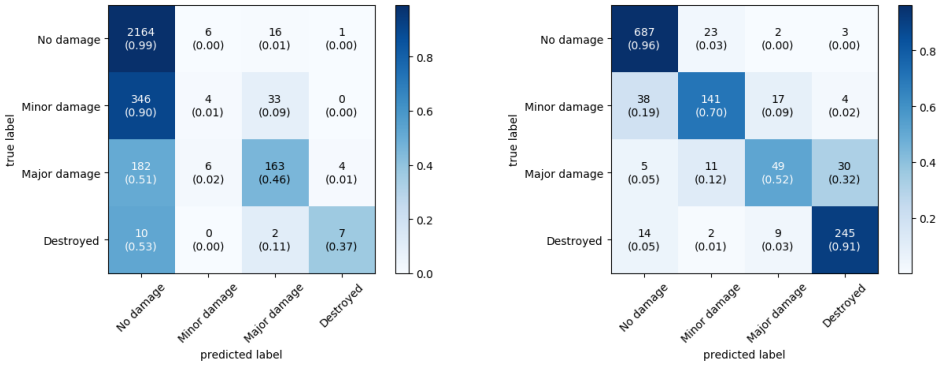


Figure 6.4: Confusion matrices of the model trained and tested on the Nepal flooding (left) and Joplin tornado (right).

makes the field appear like water. Nevertheless, the model is also making mistakes that should be recognizable, such as the building on the top right in Figure 6.5. This small qualitative analysis indicates that the model is not learning all relevant features, but also that not all data points with damage show clear damage-related features.

For Joplin, the confusion matrix shows a different pattern than for Nepal. In general, more samples are classified correctly and if they are misclassified, they are often classified as one of the adjacent classes. This is a more desirable behavior and indicates that the model is learning relevant features from the data. Again, examining a few samples helps to understand the misclassifications, see Figure 6.6. From here we can see that most misclassifications are understandable. Sometimes the model even outperforms the human annotators, such as in the bottom left where the ground truth label is clearly wrong. The top left shows minor damage, but this is difficult to differentiate due to not showing big holes or a blue tarp, which is the case for most minor damage caused by the Joplin tornado. The bottom right is wrongly predicted as destroyed, and this is probably due to the debris in front of the building. The top right is clearly mispredicted and it is ambiguous why the model mistook this building as showing no damage. One possible explanation could be that most damaged buildings show brownish damage features which is not the case here. However, if this is the explanation it indicates that the learned representation of damage features is not fully correct. Concluding, the model seems to learn most damage-related features of the Joplin tornado really well, sometimes even outperforming the human annotators.

The disadvantage of confusion matrices and qualitative analysis is that it is hard to compare a range of disasters based on them. Combining the individual numbers of all disasters and confusion matrices of Joplin and Nepal, it can be concluded that the multiclass model holds predictive value for a subset of disasters, also on the minor and major damage classes. When there is a clear distinction in the features of the damage classes with the human eye, the model seems to learn those distinctions. Concurrently, more vague and mixed imagery gives the model a hard time learning damage-related features and causes it to lean towards the majority class.

6.1.3 Multiclass grouped to binary classification

We have analyzed the performance of models trained on binary and multiclass labels. This distinction of binary versus multiclass was made because the needed granularity of predictions depends on the purpose of the damage assessment. Normally it is assumed that a model trained on binary labels is also better at distinguishing those labels. Here this assumption is tested by seeing how the multiclass model performs on distinguishing the destroyed versus other samples. This was done by training and testing the multiclass model on the four categories of labels,

Figure 6.5: Four examples of misclassified buildings in the test set of the Nepal flooding. For each building, the pre image is shown on the left and the post image on the right. The top row shows two buildings where the damage is under-predicted, while in the bottom the damage is over-predicted.

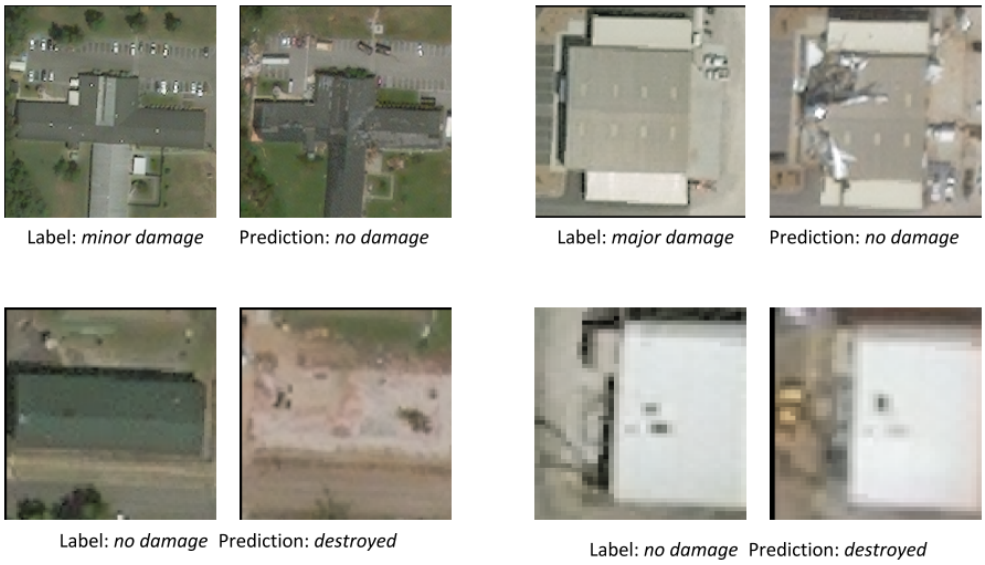


Figure 6.6: Four examples of misclassified buildings in the test set of the Joplin tornado. For each building, the pre image is shown on the left and the post image on the right. The top row shows two buildings where the damage is under-predicted, while in the bottom the damage is over-predicted.

Disaster	AUC binary model	AUC multiclass model	difference in AUC
Florence	0.496	0.874	-0.378
Sunda	0.694	0.869	-0.175
Harvey	0.853	0.897	-0.044
Matthew	0.894	0.936	-0.042
Michael	0.891	0.926	-0.035
Nepal	0.905	0.920	-0.015
Puna	0.962	0.975	-0.013
Moore	0.990	0.997	-0.007
Joplin	0.987	0.990	-0.003
Guatemala	1.000	1.000	0.000
Tuscaloosa	0.990	0.990	0.000
Palu	0.972	0.971	0.001
Midwest	0.997	0.987	0.010

Table 6.3: Binary AUC scores for a model trained on binary labels and a model trained on multiclass labels, but grouped to binary labels for performance measures. The numbers are sorted by *difference in AUC*, which is the AUC of the binary model minus the AUC of the multiclass model.

just like in the previous section, but grouping those labels and predictions to the binary classes for retrieving the performance measures. The labels were binarized by assigning the *no damage*, *minor damage*, and *major damage* classes to the *not destroyed* class and summing the probabilities for those three classes to retrieve the probability of the data point belonging to the *not destroyed* class. The purpose of this experiment is to see if a model trained on binary classes does result in a better ability to distinguish those classes. If the model trained on all four classes gives a similar or better performance than the binary model, it is favorable to solely train the multiclass model since this limits the number of model variations.

The results are displayed in Table 6.3, where for both cases the AUC is calculated on the binary distinction of *destroyed* vs *not destroyed*. As can be seen, the model trained on multiclass labels results in a better or similar AUC than when trained on binary labels. Figure 6.7 shows the distribution plots of the binary and multiclass model for the Florence hurricane, which gains the largest improvement in AUC when trained on the multiclass model compared to the binary trained model. From the figure, we can see that the ability to separate the classes is higher for the multiclass model, while it is still not great. Moreover, it can be seen that the optimal threshold is much lower than 0.5, resulting in Florence reaching the same accuracy with the binary and multiclass model. The accuracies of the different disasters are not shown here, but the pattern of the Florence hurricane is consistent. While the AUC increases or stays on the same level when switching from a model trained on binary labels to multiclass labels, the difference in accuracy between the two models is always small.

The better or similar ability of the multiclass model to separate the binary classes is surprising and the cause cannot be determined with certainty. One explanation could be that in the multiclass scenario the class imbalance is less extreme for most disasters, which causes the model to overfit less on the majority class. However, there is no relation between the change in percentage of data points belonging to the majority class and the difference in AUC, for the models trained on binary and multiclass labels. Another explanation could be that in the multiclass scenario the features per class are more distinct and thus the model is able to learn to separate those features better. Since thresholding is something that could be experimented with in the future and having fewer variations of models is favorable, it is advised to solely train the model on multiclass and group them afterwards in case only binary labels are needed. This is also the procedure applied in the remainder of the experiments.

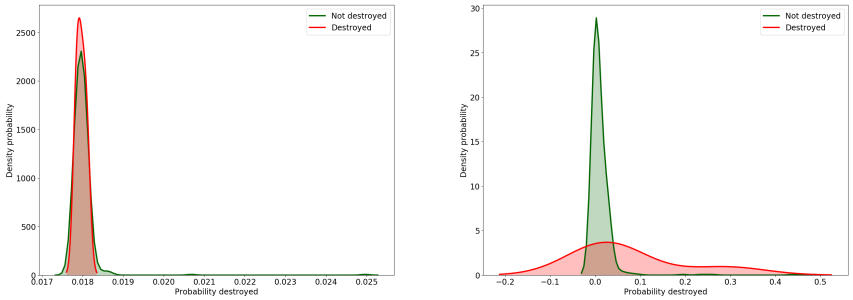


Figure 6.7: Distribution plots of the predicted probability of samples belonging to the damage class for the Florence hurricane. Trained on binary labels (left) and trained on multiclass labels and thereafter grouped to binary labels (right).

6.1.4 Understanding differences in performance

In all the experiments covered in the previous sections, it was shown that there are clear differences in performance per disaster. Trying to understand where those differences originate from is important for practical purposes. Having a better estimation of the performance of the model before or while using it in the field can make a lot of difference in usefulness, communication, and acceptance. There are many possible causes for the differences, both quantitative and qualitative. In this section, we will explore some of them.

One influence we noted before is the difference in class distributions. Figure 6.8 displays the percentage of data points belonging to a class vs the recall of that class, for each disaster and each of the four classes. From here we can see that there is a monotonically increasing, roughly logarithmic, relation between the class distribution and the recall, revealing that the recall generally increases when a larger percentage of data points belong to that class. Concurrently, we can also see from the figure that by far not all data points follow this trend.

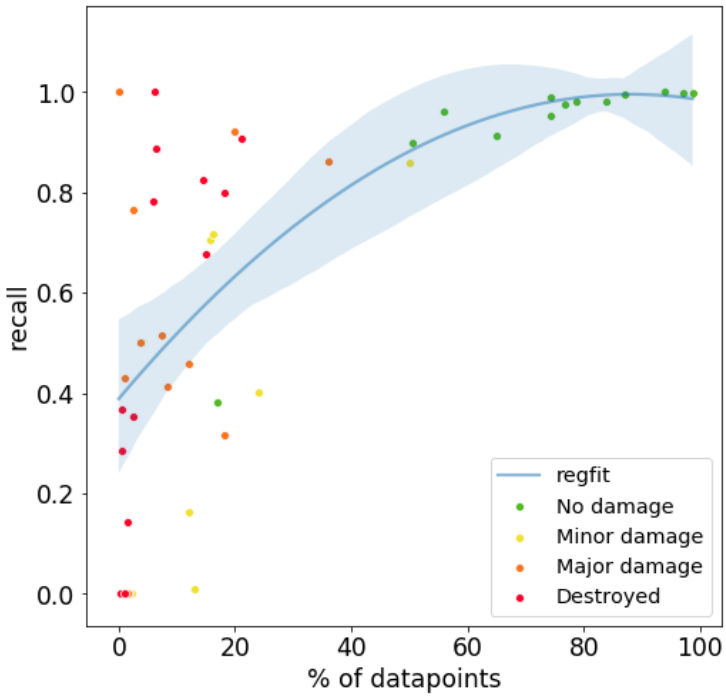


Figure 6.8: Scatter plot of the percentage of data points belonging to a class versus the recall of that class for each of the 13 tested disasters. The blue line shows the best polynomial fit and the blue area the 95% confidence interval.

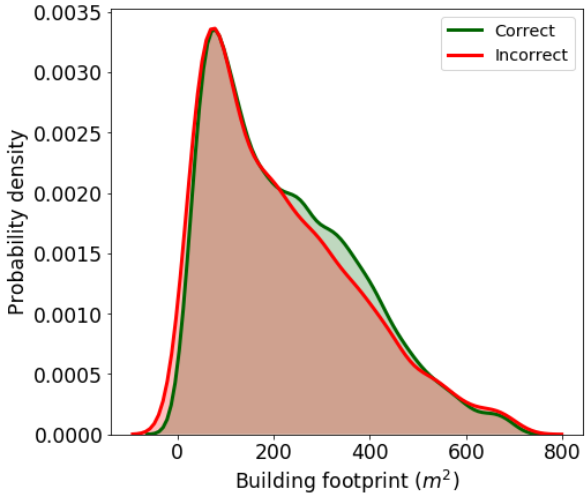


Figure 6.9: Distribution plot of the building footprint for buildings up to 700 m^2, i.e. 95% of the buildings. The green area corresponds to the distribution over correctly classified samples, and the red area corresponds to the distribution over incorrectly classified samples.

Another possible explanation for the difference in performance could be the building footprint. A hypothesis is that buildings with smaller footprints are harder to classify due to less optical clues. Figure 6.9 shows the distributions over the building footprint for the correctly and incorrectly classified buildings, for the 95% percentile of buildings. The figure shows that the distributions of the building footprint over correctly and incorrectly classified buildings largely overlap. This disproves the hypothesis, and thus apparently a larger building size is not a predictor for the chance that a building is correctly classified.

Disaster specific parameters could also be of an influence on the performance. The quantified parameters that belong to this category are the number of satellite image pairs covering a disaster, the number of buildings, the type of disaster, and the geographical region where the disaster struck. A technique to understand the relation between these parameters and the performance is by making a scatter plot of the value of the parameter versus the performance. As performance measure the AUC over the binarized labels *destroyed* and *not destroyed* was chosen since this is the measure least influenced by class imbalance. Figure 6.10 shows the scatter plots for the four parameters. From these plots, it can be seen that there is no correlation between the AUC and any of the four parameters. Nevertheless, the fact that these parameters do not have an explaining factor in the performance is also an interesting observation. For example, a negative relation between the number of image pairs covering a disaster and the performance could have been expected since more image pairs results in more variety in the data and thus the model has to learn a wider variety of features. Another interesting observation is the non-existent correlation between the number of buildings and the AUC. A smaller number of buildings means less training data and thus it could have been expected that this leads to a lower AUC.

Lastly, we can inspect parameters that are specific for a pre and post satellite image pair. These include the off-nadir angle, panchromatic resolution, Sun azimuth, Sun elevation, and target azimuth. In Figure 6.11 scatterplots for these parameters versus the AUC are shown. For each parameter a scatter plot is made for the sum of the pre and post image of that parameter, and of the absolute difference between the pre and post image of that parameter. Again, we can see that for none of the parameters, there is a very distinct relationship with the AUC. The largest influence seems to come from the sum of the Sun azimuth, the larger this sum the smaller the AUC. This same relation is somewhat visible in the difference in Sun

elevation. Both these parameters have to do with lightning and thus it can be interpreted that too little/too much lightning or larger differences in lightning can degrade performance. Surprisingly, the off-nadir angle sum and difference do not show a relationship with the AUC, while a large off-nadir angle clearly changes the appearance of the imagery and is often mentioned as a difficulty in the literature [80, 41].

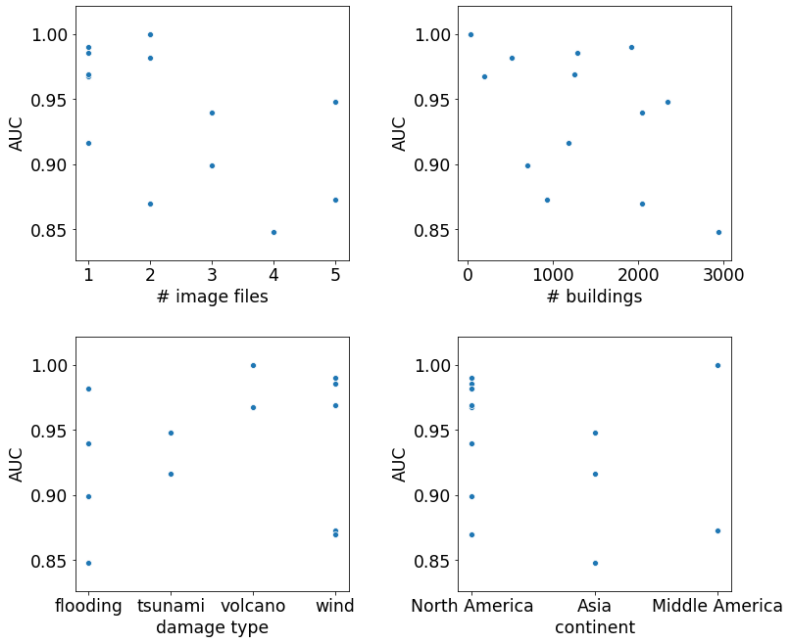


Figure 6.10: Scatter plots of the value of the parameter versus the AUC. One dot belongs to one disaster.

Section summary

- The model is able to learn damage-related features
- The performance differs per disaster
- Intermediate damage classes can be detected from aerial imagery, contrary to previous arguments
- It is preferable to train a model on a higher granularity of damage labels, even if solely a binary distinction is needed for the purpose
- There is a monotonically increasing relation between the percentage of buildings belonging to a class and the recall of that class
- Specific disaster and image parameters, such as the geographical region and the off-nadir angle, do not explain the difference in performance of the used method in this dataset

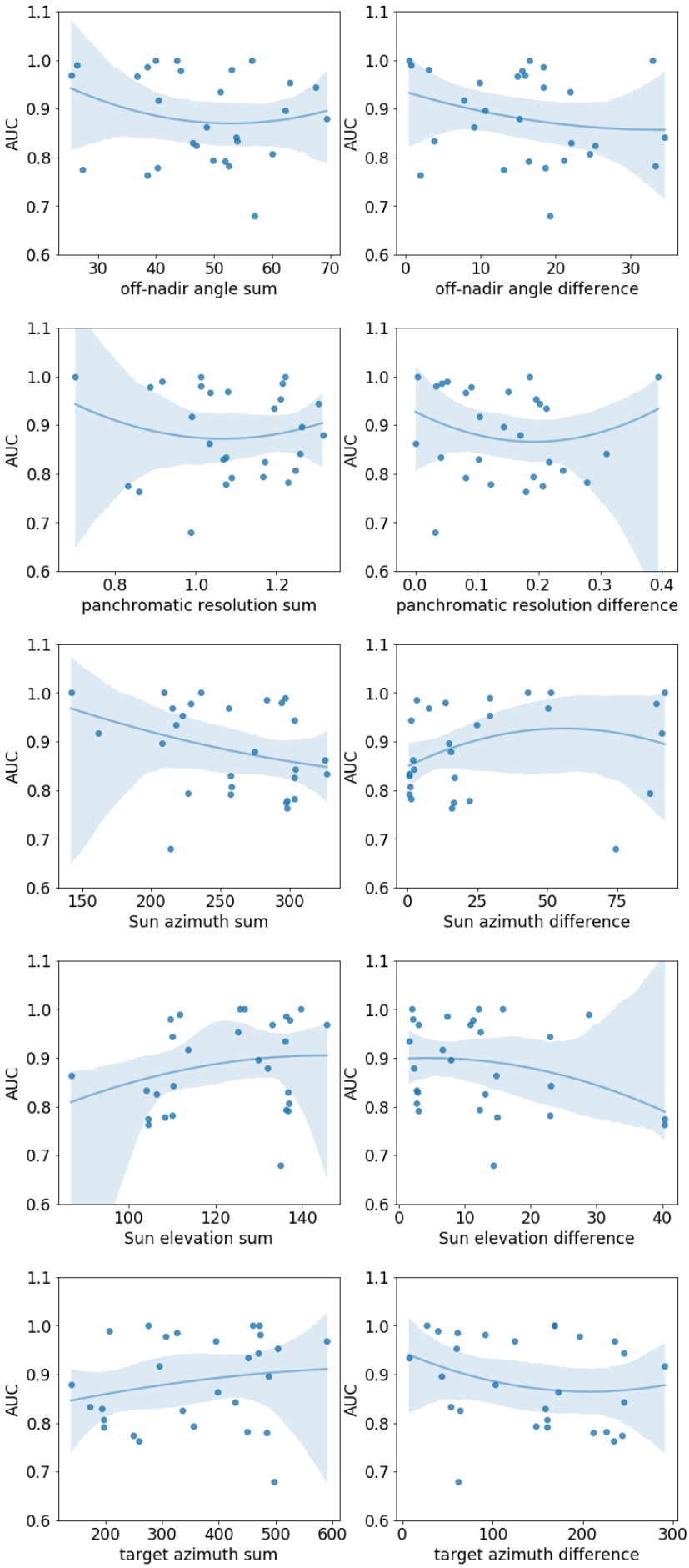


Figure 6.11: Scatters plot of the value of the parameter versus the AUC. One dot belongs to one pre-post satellite image pair. The x-axis of the left column represents the sum of the parameter over the pre and post image. In the right column, the x-axis represents the absolute difference in the parameter value between the pre and post image.

6.2 TESTING ON A DISASTER WITHOUT LABELED DATA

The experiments on training and testing on the same disaster gave insights into the predictive value of the model and the differences between disasters. However, for the model to have actual value in practice, it should be able to predict well on a disaster for which no labeled data is available since this is often the practical data setting. In our set-up, this translates that the test disaster should not be included in the training set. This is a form of transfer learning where the source domain is not the same as the target domain, i.e. $\mathcal{D}_s \neq \mathcal{D}_t$ (see Section 3.3). Though most research on transfer learning assumes partial knowledge of the target domain, i.e. a subset of data points, we do not, since this is not the optimal case for a quick disaster response. One advantage we do have is that we have the full information of several source domains, i.e. several disasters, which can be used during training.

To determine the transferability of disasters, experiments were done to discover if training on a (set of) disaster(s) which does not include the test disaster, can reach performance similar to a model that is trained on the test disaster. The proposition here is that the maximum performance possible, given the model architecture and the test disaster, is that of a model trained and tested on this *test disaster*. Thus, the goal is to get as close as possible to this performance, without including the test disaster in the training set.

This experiment is done for two test disasters and purposefully two very different disasters were chosen. One is the Joplin tornado, which struck in the USA, caused wind damage, and performed rather well when trained on itself as we saw in the previous section. The other is the Nepal flooding, which took place in Asia and performed rather poorly when trained on itself.

For both test disasters, the structure was the same. The set of test buildings was kept consistent, namely 10% of one of the two disasters, and different training sets were created according to the following set-up:

1) One disaster with the same damage type as the test disaster. This was repeated for all available disasters with this damage type.

2) One by one add disasters of the same damage type, in the order of ascending macro F1 from 1).

3) All disasters of 2) plus a disaster of a different type.

4) A wide mix of disasters.

5) The same disaster as the test disaster

The goal is to compare the performance of step 5) to all the other steps, to see what the gap is between the maximum performance and the realistic production performance, where the model is not trained on the test disaster.

6.2.1 Joplin tornado

In Table 6.4 the results of testing on the Joplin tornado according to the different steps of training sets described above are shown. The macro F1, AUC on the binarized classes, and recall per class were chosen as performance measures. For set-up 1), where the model is only trained on one disaster, we can see that the performance differs per training disaster. For three out of four disasters, the performance measures show that the models learn damage related features that partially transfer to the Joplin tornado. The Moore tornado is the disaster with the highest performance, reaching 80% of the macro F1 compared to being trained on the test disaster itself. The other surprising result is that when training on hurricane Matthew, the performance is bad. What is the cause of this is debatable, but one source might be the completely different class distribution. As can be seen from the recall scores,

	Train disasters					Recall			
Step	Names	Types	Regions	macro F1	AUC	No dam.	Minor	Major	Destr.
1)	Moore	tornado (1)	North America (1)	0.627	0.981	0.993	0.425	0.368	0.559
	Tuscaloosa	tornado (1)	North America (1)	0.597	0.982	0.994	0.410	0.368	0.496
	Michael	hurricane wind (1)	North America (1)	0.583	0.954	0.994	0.395	0.358	0.467
	Matthew	hurricane wind (1)	Central America (1)	0.225	0.667	0.206	0.775	0.516	0.007
2)	Moore, Tuscaloosa	tornado (2)	North America (2)	0.683	0.983	0.987	0.515	0.411	0.659
	Moore, Tuscaloosa, Michael	tornado (2), hurricane wind (1)	North America (3)	0.715	0.985	0.979	0.550	0.463	0.730
	Moore, Tuscaloosa, Michael, Matthew	tornado (2), hurricane wind (2)	North America (3), Central America (1)	0.733	0.984	0.976	0.585	0.589	0.700
3)	Moore, Tuscaloosa, Michael, Harvey	tornado (2), hurricane wind (2), hurricane flood (1)	North America (4), Central America (1)	0.717	0.985	0.987	0.530	0.537	0.704
4)	Michael, Florence, Harvey, Midwest, Guatemala, Matthew, Palu	hurricane wind (2), hurricane flood (2), flood (1), tsunami (1), volcano (1)	North America (4), Central America (2), Asia (1)	0.485	0.963	0.989	0.075	0.116	0.807
5)	Joplin	tornado (1)	North America (1)	0.787	0.990	0.961	0.705	0.516	0.907

Table 6.4: Performance of models trained on different sets of disasters. The test set is in all cases 10% of the Joplin tornado. The orange row is also trained on the Joplin tornado.

this model is bad at recognizing *no damage* and indeed hurricane Matthew is the only disaster where *no damage* is not the majority class, see table 6.2. The other cause might be that the buildings and their damage look quite different compared to those of the Joplin tornado. This different appearance can be caused by the fact that the hurricane took place in another part of the world, where building typology might be different, that it was a hurricane instead of a tornado, or due to the image parameters.

When adding more disasters of the same damage type as the Joplin tornado, i.e. wind, the performance in terms of macro F1 continues to improve (step 2). When adding a disaster of another type of damage, in this case flood, the macro F1 degrades slightly (step 3). Training on a mix of all types of damage, leads to a drop in macro F1, while still having a high recall for *no damage* and *destroyed*.

From these results, we can conclude that for the case of the Joplin tornado, the model can generalize well to other domains, reaching 93% of the macro F1 compared to when trained on the Joplin data. This performance is reached when training on all the other disasters with wind damage in the dataset. Thus, adding more disasters of the same type improves the ability to generalize, and consequently, the differences between disasters with the same damage type in the training set, work as an advantage.

The best performing set of training disasters includes the Tuscaloosa tornado, Moore tornado, Michael hurricane, and Matthew hurricane, and is from here on

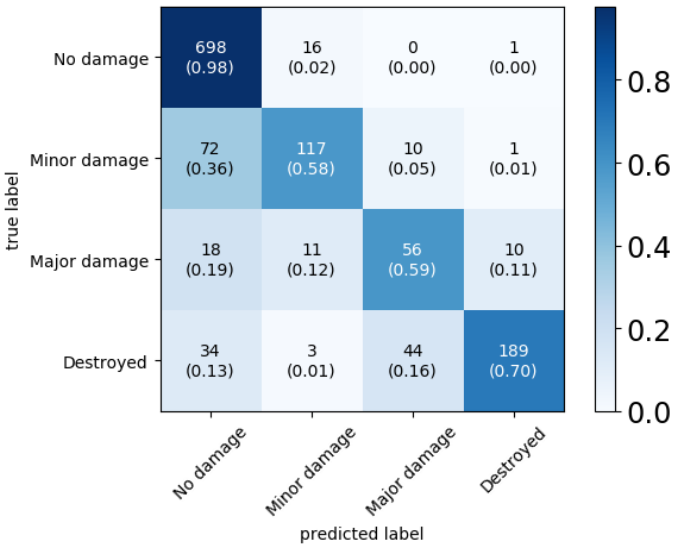


Figure 6.12: Confusion matrix of the model trained on four wind disasters and tested on the Joplin tornado.

Label: *major damage*
Prediction trained on Joplin: *destroyed*
Prediction trained on four wind: *major damage*

Label: *destroyed*
Prediction trained on Joplin: *destroyed*
Prediction trained on four wind: *no damage*

Figure 6.13: Two buildings before and after the Joplin tornado, and the predictions made by the model trained on the Joplin tornado and the model trained on four wind disasters, not including the Joplin tornado.

referred to as the *four wind disasters*. To better understand the workings of the model trained on this data, we can look at the confusion matrix, as displayed in Figure 6.12. The matrix shows that the model learns to recognize damage well and that the main loss in performance comes from under-predicting damage.

The predictions of the model trained on the *four wind disasters* and trained on the Joplin tornado, match on the largest fraction of buildings. Nonetheless, it is interesting to qualitatively examine buildings they do not agree on. The main disagreement is on cases where the model trained on Joplin estimates the damage at a higher level than the model not trained on Joplin, see Figure 6.13 for two examples. The left example shows a building where the model trained on four wind disasters correctly classifies the building as *major damage*, while the model trained on Joplin overestimates the damage as *destroyed*. The border between the two classes here is ambiguous, but it shows that the Joplin model judges the same damage differently. The right building is *destroyed* and correctly classified by the Joplin model, while the model trained on four wind disaster classifies it as *no damage*. This is clearly wrong, but might be influenced by the fact that it is not the clearest picture of damage. The color is clearly different, but the structure appears relatively the same after the disaster as before, thus this might be a display of damage that does not occur for the *four wind disasters*.

To further understand the performance of the *four wind disasters* model, this model can be tested on 100% of the Joplin data and then visualized spatially. Figure 6.14 shows this visualization for a part of the damaged area. Here we can see that most

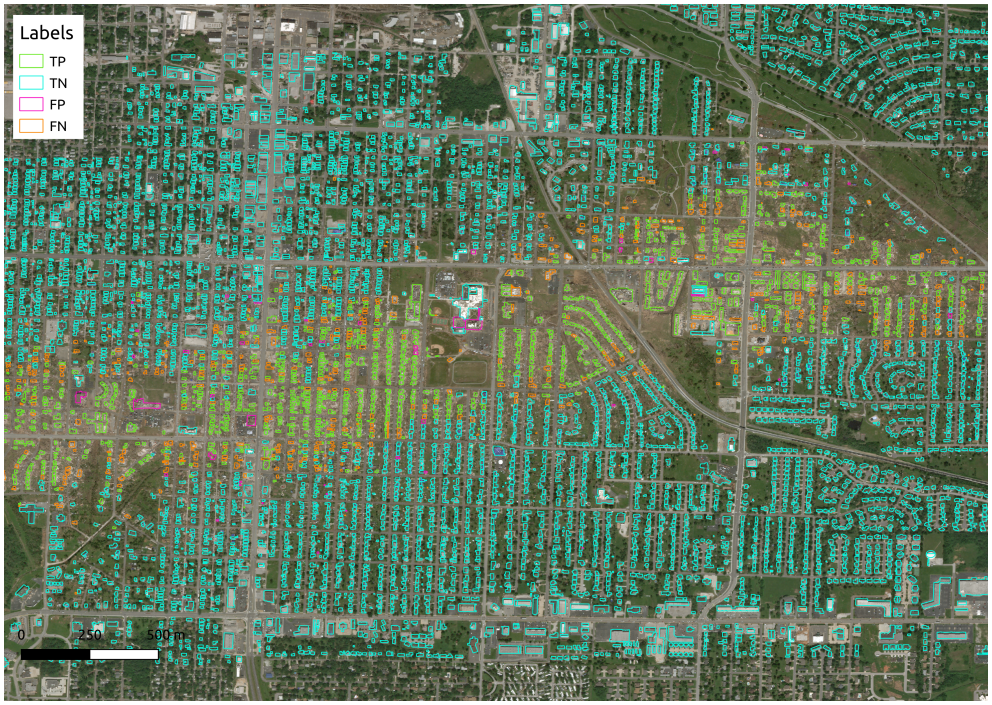


Figure 6.14: The TP, TN, FP, and FN predictions on all buildings of the Joplin tornado by the model trained on a mix of four disasters with wind damage, where a *positive* data point belongs to the *destroyed* class, and a negative data point to the *not destroyed* class. The area shown is a smaller part of the whole affected area.

incorrect predictions are on the edge of the path of the tornado, which is logical since in those areas the looks of the damage and their assigned labels are the most ambiguous.

6.2.2 Nepal flooding

The Nepal flooding is a completely different disaster and thus is a good test case to see if the results of the Joplin tornado are transferable to other disasters. Table 6.5 shows the same type of experiment as was performed for the Joplin tornado, but now with the Nepal flooding as the test disaster. For set-up 1), where the model is trained on one disaster, we can see that the performance differs per disaster, which is a similar pattern as for the Joplin tornado. On the contrary, the performance of those individual disasters is worse. Where for the Joplin tornado a performance of 80% of the original macro F1 was reached, for the Nepal flooding this is merely 53%. Judging by the AUC, the performance is even worse, where only the model trained on the Harvey hurricane has some predictive value, indicated by the AUC being larger than 0.5, while its AUC of 0.732 is still rather low. From the table it can be seen that the models trained on the Midwest flooding and Florence hurricane, only recognize *no damage* and *major damage*. When looking at the results of those models when tested on the disasters themselves (see Table 6.2), we see similar behavior, which might indicate that the model is only able to learn information on those two classes based on the input data. Surprisingly, the behavior of overpredicting on *major damage* is also observed for the Harvey Hurricane when tested on itself, but disappears when tested on the Nepal flooding. Why this is the case, is not understood.

When combining the disasters with flood damage in the training set (step 2), the performance does not improve compared to the maximum performance of the individual disasters. This is different behavior than was observed for the Joplin tornado, where performance improved when adding more disasters of the same

	Train disasters					Recall			
Step	Names	Types	Regions	AUC	macro F1	No dam.	Minor	Major	Destr.
1)	Midwest	flood (1)	North America (1)	0.258	0.395	0.898	0.000	0.211	0.000
	Harvey	hurricane flood (1)	North America (1)	0.238	0.732	0.985	0.003	0.023	0.053
	Florence	hurricane flood (1)	North America (1)	0.173	0.480	0.382	0.013	0.504	0.000
2)	Midwest, Harvey	flood (1), hurricane flood (1)	North America (2)	0.221	0.568	0.910	0.000	0.054	0.000
	Midwest, Harvey, Florence	flood (1), hurricane flood (2)	North America (3)	0.217	0.711	0.995	0.000	0.008	0.000
3)	Midwest, Harvey, Florence, Palu	flood (1), hurricane flood (1), tsunami (1)	North America (3), Asia (1)	0.227	0.653	0.978	0.000	0.006	0.105
4)	Michael, Florence, Harvey, Midwest, Guatemala, Matthew, Palu	hurricane wind (2), hurricane flood (2), flood (1), tsunami (1), volcano (1)	North America (4), Central America (2), Asia (1)	0.226	0.657	0.936	0.031	0.017	0.000
5)	Nepal	flood (1)	Asia (1)	0.482	0.920	0.989	0.010	0.459	0.368

Table 6.5: Performance of models trained on different sets of disasters. The test set is in all cases 10% of the Nepal flooding. The orange row indicates the model that is also trained on the Nepal flooding.

damage type. For an unknown reason, the model with more disasters loses its ability to recognize the *major damage* class. When adding more disasters of different damage types (steps 3 and 4), the model does not yield a better performance than when trained on one disaster either.

In an attempt to understand the disappointing performance better, Figure 6.15 shows the confusion matrix for the model with the highest macro F1, reached when trained solely on the Midwest flooding. The matrix shows that the model is only predicting *no damage* and *major damage*. Visualzing the predictions made by this model when tested on 100% of the Nepal flooding, see Figure 6.17, shows that even clearly damaged areas are not classified correctly. Visually inspecting some of the mispredicted buildings, does not show clear indications for the reasons of mispredictions. As an example, Figure 6.16 shows two mispredicted buildings that clearly show their respective classes and thus indicate that the model does not learn the relevant damage-related features.

This behavior might indicate that the model is simply not able to learn well enough to distinguish the different classes based on the information that is contained in the data of the Midwest flooding. Especially since the inability of correctly predicting data points, even though they show clear characteristics of the respective class, is also observed when tested on the Midwest flooding itself. The overall disappointing predictive behavior on the Nepal flooding of all the combinations of training disasters, both in terms of macro F1 as AUC, could be caused by the fact that the data characteristics of the Nepal flooding are quite different from the other disasters. The imagery has, for example, a high off-nadir angle compared to the other disasters, resulting in a different appearance of buildings. Moreover, the buildings look different due to being in a different part of the world. Lastly, the flood characteristics can be different, where for example the flooded water for the Nepal flooding is green, while for the Midwest flooding it is brown.

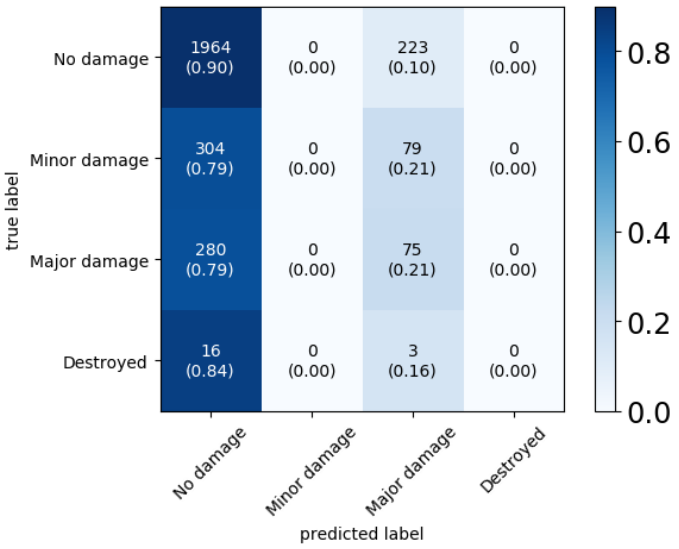


Figure 6.15: Confusion matrix of the model trained on the Midwest flooding and tested on 10% of Nepal.

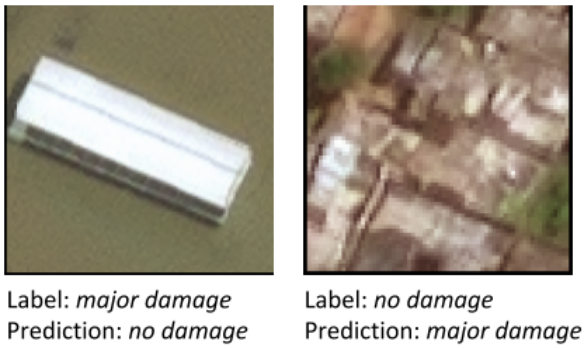


Figure 6.16: Two examples of buildings on the post disaster imagery that show clearly their true label, but that are wrongly classified by the model trained on the Midwest flooding and tested on the Nepal flooding.

What exactly causes the fact that none of the training sets performs well on the Nepal flooding as the test disaster, remains to be researched further. With the current results, the performance for the Nepal flooding does not seem good enough for practical purposes, unlike the results from the Joplin tornado. For the Nepal flooding, it was shown that the best result is gained with training on one disaster. However, the specific features associated to damage in a new disaster, as well as the building and environmental characteristics (e.g. water color), are not known a priori and thus it is not known which of the individual disasters would result in the best performance. Thus, it seems to be a reasonable choice to still utilize the model trained on all disasters of the same damage type, since this gives a similar level of performance as when trained on the best individual disaster, while providing more certainty on the level of this performance.

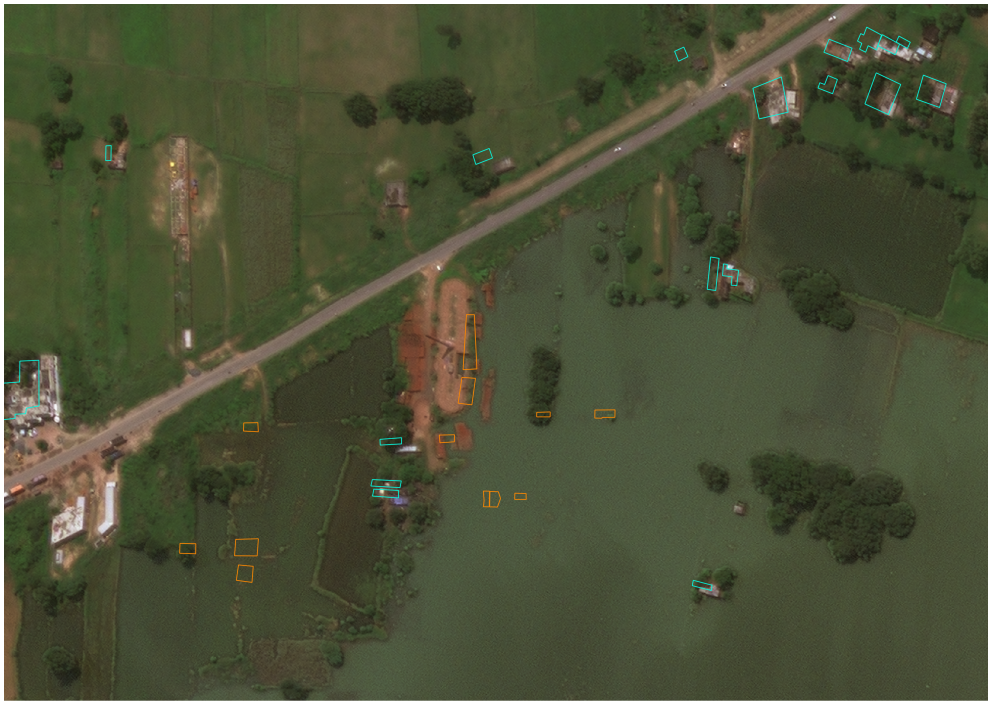


Figure 6.17: The TP, TN, FP, and FN predictions on all buildings of the Nepal flooding by the model trained on the Midwest flooding, where a *positive* data point belongs to the *destroyed* class, and a negative data point to the *not destroyed* class. The area shown is a smaller part of the whole affected area.

Section summary

- The performance on a disaster that was not included in the training set highly depends on the disaster
- For the two disasters tested here, 93% and 53% of the macro F1 when trained on the test disaster was reached
- The best performance is reached by solely training on disasters with the same damage type, e.g. wind or flood, as the test disaster

6.3 MODIFICATIONS TO THE MODEL

We have seen that the performance of the model and the transferability of learned weights to other disasters differs per disaster. To discover if the general performance and generalizability could be improved, several modifications to the model were made. A subset of these modifications add a feature to the model, others simplify the model in order to see if there was any effect of those parts of the model. These modifications are tested on four combinations of training and testing disasters. Joplin and Nepal are selected as the test disasters. For both disasters the modifications are tested when trained on that disaster, and when trained on the set of disasters that gained the highest macro F1 when not trained on the test disaster, as defined in the previous section. These are the Midwest flooding as training set when tested on the Nepal flooding, and the combination of the Moore tornado, Tuscaloosa tornado, Michael hurricane, and Matthew hurricane, referred to as the *four wind disasters*, when tested on Joplin. Before implementing any modifications, another run of each of the four settings is done to test the stability. Thereafter, a set of modifications are tested that are related to imbalance, data augmentation, and model architecture.

6.3.1 Another run

To get a better idea of the stability of the performance the models without modifications were trained again, referred to as *run 2*. The results for each of the four set-ups are shown in Table 6.6. The results show that the difference in macro F1 between the two runs ranges from 0.001 to 0.038, depending on the train and test disasters. Since these differences are not that large, it seems the model finds approximately the same optimum. On the other hand, the differences between experiments in previous sections were also very small, and thus the differences between the runs might still have a significant impact. Hence, in the coming sections where the modifications to the model are tested, these changes are compared to the performance measures of both runs of the model without changes. If the modification results in a better score than both runs, there is more certainty that the modification actually gives added value.

					Recall			
Train disaster(s)	Test disaster	Experiment	Macro F1	AUC	No dam.	Minor	Major	Destr.
Joplin	Joplin	run 2	0.786	0.987	0.917	**0.750**	**0.632**	0.893
		run 1	**0.787**	**0.990**	**0.961**	0.705	0.516	**0.907**
4 wind	Joplin	run 2	0.696	**0.985**	**0.980**	0.530	**0.684**	0.578
		run 1	**0.733**	0.984	0.976	**0.585**	0.589	**0.700**
Nepal	Nepal	run 2	0.463	**0.933**	0.984	**0.052**	**0.501**	0.158
		run 1	**0.482**	0.920	**0.989**	0.010	0.459	**0.368**
Midwest	Nepal	run 2	0.220	0.196	**0.980**	0.000	0.020	0.000
		run 1	**0.258**	**0.395**	0.898	0.000	**0.211**	0.000

Table 6.6: The results of running the same experiment twice for four different set-ups. The results per set-up are sorted by ascending macro F1.

6.3.2 Imbalance

The class imbalance in the data causes difficulty for the model to predict the minority classes correctly, as was shown in Sections 6.1 and 6.2. With the goal of limiting the decrease in performance due to the imbalance, resampling and cost-sensitive learning were implemented, as described in Section 5.7.1. These methods are here referred to as *resample* and *weighted loss*. Table 6.7 shows the results of these modifi-

cations for the four different set-ups, plus the results of the original runs as a comparison. As can be seen, it differs per set-up if the resample and/or weighted loss modification improves the performance compared to the original runs. Nonetheless, in all cases we can observe that the modifications result in better recall over the damage classes. This is an important criterion since recognizing the damaged buildings is often more crucial than recognizing the none damaged buildings. Figure 6.18 shows the relation between the recall and class percentage per class. Here, it can be seen that while resampling and weighted loss diminish the relation between the recall and class percentage, a dependency does still exist. Nevertheless, since the performance of the damage classes is improved when implementing these modifications, it is recommended to use one of these methods. The results of both methods are very close to each other, and thus which one to choose is almost arbitrary. If one has to be chosen, the more consistent AUC when using resampling can be a reason to choose the *resampling* over *weighted loss*.

Train disaster(s)	Test disaster	Experiment	Macro F1	AUC	Recall			
					No dam.	Minor	Major	Destr.
Joplin	Joplin	run 2	0.786	0.987	0.917	**0.750**	0.632	0.893
		run 1	0.787	**0.990**	**0.961**	0.705	0.516	0.907
		resample	0.790	**0.990**	0.923	0.745	0.621	**0.926**
		weighted loss	**0.791**	0.989	0.923	**0.750**	**0.705**	0.867
4 wind	Joplin	run 2	0.696	**0.985**	**0.980**	0.530	0.684	0.578
		resample	0.730	**0.985**	0.936	**0.635**	**0.747**	0.685
		run 1	0.733	0.984	0.976	0.585	0.589	0.700
		weighted loss	**0.735**	0.983	0.968	0.615	0.684	**0.689**
Nepal	Nepal	weighted loss	0.409	**0.933**	0.757	0.243	0.501	0.474
		run 2	0.463	**0.933**	0.984	0.052	0.501	0.158
		resample	0.474	0.862	0.709	**0.394**	**0.606**	**0.526**
		run 1	**0.482**	0.920	**0.989**	0.010	0.459	0.368
Midwest	Nepal	resample	0.213	**0.678**	0.504	**0.386**	0.037	**0.053**
		run 2	0.220	0.196	**0.980**	0.000	0.020	0.000
		weighted loss	0.225	0.170	0.975	0.013	0.017	0.000
		run 1	**0.258**	0.395	0.898	0.000	**0.211**	0.000

Table 6.7: The results of implementing resampling and weighted loss for four different set-ups of train and test disasters. The results per set-up are sorted by ascending macro F1.

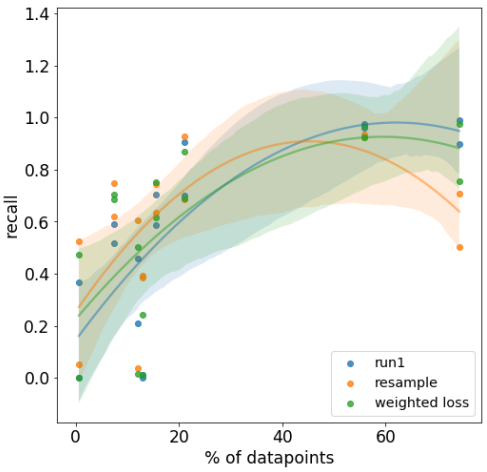


Figure 6.18: Scatter plot of the class percentage versus the recall for three versions of the model: the original version, resample, and weighted loss.

6.3.3 Data augmentation

Data augmentation is often mentioned as an important pipeline to prevent overfitting and thus should improve performance to a new dataset. Three versions of augmentation schemes were tested, as elaborated upon in Section 5.7.2. The original scheme, a scheme adapted from [50], referred to as *paper aug*, and not applying any augmentation, referred to as *no aug*. The results of these three versions are displayed in Table 6.8, where *run 1* and *run 2* use the original scheme. From here it can be seen that the results differ per set-up. No augmentation, in general, scores the lowest, except when training and testing on *Nepal*. There is no real logical explanation for this. When examining the training loss and macro F1 on the validation set during training, the expected behavior is shown where the differences in macro F1 are very small between the different set-ups. It might thus solely be chance, but in all cases the difference between the original augmentation scheme and not applying augmentation is small. Augmentation might have been expected to have a larger influence since it is normally an integral part of any neural network. However, the small but positive effect of augmentation is consistent with previous research on automated building damage assessment [49, 50, 51].

Whether applying the original augmentation scheme or the augmentation scheme of [50] is more advantageous depends on the set-up. The differences are rather small, which can be expected since the two augmentation schemes are similar.

Combining all the results, it is advised to apply augmentation since it results in a small improvement. Which precise augmentation scheme is the best, depends on the set-up, but the differences are rather small so both are suitable.

Train disaster(s)	Test disaster	Experiment	Macro F1	AUC	Recall			
					No dam.	Minor	Major	Destr.
Joplin	Joplin	no aug.	0.759	0.983	0.948	0.695	0.400	**0.948**
		paper aug.	0.772	0.988	0.959	0.720	0.453	0.904
		run 2	0.786	0.987	0.917	**0.750**	**0.632**	0.893
		run 1	**0.787**	**0.990**	**0.961**	0.705	0.516	0.907
4 wind	Joplin	no aug.	0.670	0.971	0.962	**0.600**	0.316	**0.715**
		run 2	0.696	**0.985**	**0.980**	0.530	**0.684**	0.578
		run 1	0.733	0.984	0.976	0.585	0.589	0.700
		paper aug.	**0.738**	0.982	0.972	**0.600**	0.632	0.707
Nepal	Nepal	paper aug.	0.448	0.915	0.970	0.050	**0.589**	0.158
		run 2	0.463	**0.933**	0.984	0.052	0.501	0.158
		run 1	0.482	0.920	**0.989**	0.010	0.459	**0.368**
		no aug.	**0.523**	0.913	0.902	0.219	0.544	0.316
Midwest	Nepal	no aug.	0.215	0.280	**1.000**	0.000	0.003	0.000
		run 2	0.220	0.196	0.980	0.000	0.020	0.000
		paper aug.	0.223	0.271	0.963	0.000	0.034	0.000
		run 1	**0.258**	**0.395**	0.898	0.000	**0.211**	0.000

Table 6.8: The results of using no augmentation scheme (*no aug.* and using the scheme of [50] (*paper aug.*). The results per set-up are sorted by ascending macro F1.

6.3.4 Model architecture

The model architecture is defining for performance, but also for computational costs and data requirements. To loosen those, two downgrades of the complexity of the architecture were done. In the first one, only post imagery was used, which is referred to as *post*. In the second set-up, both pre and post imagery were used, but the weights of both CNNs were shared, referred to as *shared*. For both modifications, the model has to be retrained since the importance of different weights will change. The results are displayed in Table 6.9.

From the table, we can see that only using post imagery never improves the performance compared to both runs of the original architecture. How much the performance degrades depends on the disaster, but in general the differences are smaller than one might expect. Two papers have done a similar experiment where [49] did not find any significant difference while [42] did find a significant difference. This makes it even more clear that the performance is highly situation-dependent. Based on these experiments it is advised to use the pre imagery if it is easily available, but they also show that solely using post imagery might be a suitable alternative if the pre imagery is not available or has very low data quality.

For the architecture where the weights are shared, the conclusion again depends on the specific situation. In three out of four cases the macro F1 is larger when not sharing the weights, whereas for training it on the four wind disasters and testing on Joplin the macro F1 is larger when the weights are shared. However, the AUC never increases when sharing the weights. Since the train and test computation time did not significantly decrease when using shared weights, it can be concluded that it is beneficial to stick to not sharing the weights.

Train disaster(s)	Test disaster	Experiment	Macro F1	AUC	Recall			
					No dam.	Minor	Major	Destr.
Joplin	Joplin	post	0.745	0.989	**0.969**	0.700	0.305	**0.933**
		shared	0.746	0.986	0.890	**0.760**	0.432	0.941
		run 2	0.786	0.987	0.917	0.750	**0.632**	0.893
		run 1	**0.787**	**0.990**	0.961	0.705	0.516	0.907
4 wind	Joplin	run 2	0.696	**0.985**	**0.980**	0.530	**0.684**	0.578
		post	0.696	0.984	0.973	0.575	0.611	0.589
		run 1	0.733	0.984	0.976	0.585	0.589	0.700
		shared	**0.742**	0.983	0.971	**0.595**	0.432	**0.859**
Nepal	Nepal	shared	0.315	0.843	**0.995**	0.000	0.197	0.053
		post	0.366	0.860	0.989	0.000	0.493	0.000
		run 2	0.463	**0.933**	0.984	**0.052**	**0.501**	0.158
		run 1	**0.482**	0.920	0.989	0.010	0.459	**0.368**
Midwest	Nepal	run 2	0.220	0.196	0.980	0.000	0.020	0.000
		post	0.223	0.103	**0.996**	0.000	0.020	0.000
		shared	0.237	0.224	0.960	0.000	0.073	0.000
		run 1	**0.258**	**0.395**	0.898	0.000	**0.211**	0.000

Table 6.9: The results of using solely post imagery and sharing weights for four different set-ups of train and test disasters. The results per set-up are sorted by ascending macro F1.

Section summary

- Applying resampling or a weighted loss based on class frequency, improves the performance on minority classes
- The effects of data augmentation are smaller than expected, but it is still recommended to implement an augmentation scheme
- Solely using post imagery, compared to also including pre imagery, results in small drop in performance
- Weight sharing of the two CNNs does not improve performance and does not significantly lower computation time, and thus it is advised to keep separate weights.

7 DISCUSSION

In this thesis, several important facets of automated building damage assessment and its applicability to real-world scenarios were discovered. At the same time, it became clear that there are still many more steps to take to fully utilize the potential of automated building damage assessment. This chapter discusses several directions for future work, and concludes with an ethical discussion on the potential application of the automated building damage assessment tool.

The focus of future work should be two-fold. First, it should be further researched what the requirements of the model are for it to match the user needs, which can only be done through an iterative design process in collaboration with potential users. Concurrently, the performance of the model should be further understood and improved in the context of real-world situations, by the means of further exploring machine learning and data processing techniques.

7.1 USER NEEDS

Value is not created by focusing on optimal performance, but rather by focusing on what is necessary and useful, which is often overlooked by researchers. Accordingly, it is important to know which needs automated damage assessment from aerial imagery can fulfill and which it cannot. During the course of this thesis, a first attempt was made to identify the users and their needs (Section 2.2) and a human-centered design (HCD) process within 510 was started. These steps again clarified how important and complex of a task defining the real value is. To create value, firstly it should be defined further for which response actions automated damage assessment can be useful. A useful assessment provides the necessary information at the right time and with good enough accuracy. What the "necessary information", "right time" and "good enough accuracy" are, depends on the user requirements. These user requirements also translate to model requirements such as the damage class granularity, spatial granularity (building vs area), and available data when the assessment is to be made. Likewise, the user needs influence the choice of the performance measure. For some purposes, it can be more important not to overestimate damage, for example for search and rescue, while for others it is essential not to miss any damaged buildings, such as for cash assistance.

Regarding the damage class granularity, it should be decided which division of classes there should be. This can be the four classes or binary divisions used in this research, or another damage scale. Moreover, it could also be decided to use regression instead of classification, as was done in the previous development of this model. This can be beneficial since damage is more of a continuum, which is better captured by regression. However, it also holds disadvantages such as needing well-labeled data and the ambiguity of which continuous values of damage then require which humanitarian response. Which output type fits best, has to be discovered by the iterative design process together with potential users.

Moreover, the automated damage assessment tool should not be seen as an isolated product, but as a tool that will complement current methods. The generated damage information should be combined with other sources of information. This information can be related to the buildings, such as building topology, building purpose, and an indication of buildings that are not in use at the moment. This

building information should also be coupled with other types of information on the vulnerability and coping capacity of the impacted population, since this is essential to create a good vision of the needed aid. Besides the produced information, the presentation and communication of the results are of high importance, as is the process of publishing them given the many actors involved in humanitarian aid. All these factors have to be further researched and designed before the full potential of the automated damage assessment tool can be reached.

7.2 THE MODEL

Two of the most interesting results discovered in this research are the difference in performance and the difference in transferability between disasters. To further understand these results, four points need to be considered. Firstly, all experiments should be run with cross-validation to ensure consistency across runs and data [81]. This was not done in this research due to the limited time. Secondly, a deeper analysis should be conducted on the origins of the differences in performance between disasters, also examining qualitative differences. Thirdly, more experiments with testing on disasters that the model was not trained on would be beneficial. This should help to answer questions such as "Is flood damage harder to classify than wind damage when not trained on the test disaster? Does including previous disasters of the same geographical area improve performance? Or does the transferability depend on the specific characteristic of the disaster?". Lastly, if the model is applied to areas for which little or no data was available in this research, it is advisable to include and test on data of these areas. Due to different appearances of buildings and environments, the model might perform differently and it is important to understand how so.

As stated in Section 7.1, the model should fit the user needs. An important aspect of this is how performance is measured. This applies to the loss function, and the measures evaluating the performance on the validation set and test set. An interesting approach to explore is cost-sensitive learning, where a cost matrix for the different misclassifications can be defined, and used for the loss function [76] as well as for the evaluation on the validation and test sets.

Another potential approach to make the classification better fit the purpose is by experimenting with the classification threshold after training the model. In the current setting, the model is underpredicting damage, which is not optimal for several real-world applications. The experiments showed that for a set of disasters this is because classifying the data point to the class with the largest probability does not always lead to the most accurate separation of classes. Therefore, trying out different thresholding methods could hold potential for obtaining a more equal recall of the different classes [74]. The chosen thresholds could be adjusted to the purpose of the assessment. Lastly, without changing the method of measuring performance, making more use of the outputted probabilities could be beneficial for interpreting the results, for example, to provide a certainty interval of the classifications.

Naturally, there are numerous points that can be changed in the model architecture and its settings. The first suggestion would be a simpler architecture with fewer weights. It was shown in this research that the current architecture is able to learn damage-related features. However, computational costs are rather high, and previous research has mostly been based on simpler models that showed good performance as well. When implementing a smaller model, it is advisable to decrease the input size of the observations. Currently, all images are resized to 299×299 pixels, while the original size of most images is smaller. Furthermore, it might decrease performance to use the same method of creating the bounding box and resizing for all buildings regardless of each building's size. While there was no significant difference in the model's ability to classify smaller or larger buildings, spatial dimensions, given this method of bounding boxes, are not consistent across

building sizes since the original bounding boxes have different sizes but are all resized to the same dimensions. An interesting idea would be to use two inputs of the building as done by Fujita et. al. [49], one where a bounding box around the building is drawn whereafter this bounding box is resized to a given size, while the other input is a patch of the image with the given building in its center but not rescaled based on the building size. With this method, smaller buildings get more of the surrounding environment as input which might be beneficial, especially for floodings where it was shown, by qualitative analysis, that sometimes the damage features can be seen clearly only if a larger area around the building is included.

In addition, data augmentation holds interesting opportunities. While this research showed that the impact of the current augmentation scheme was small, other augmentations might have a larger influence. For example, this study showed that extreme levels of lightning have a slightly negative impact. Thus, one could experiment with techniques to compensate for this effect, for example through histogram equalization.

It was further observed that class imbalance plays a large role in model performance. The implemented weighted loss and balanced resampling diminished these effects, but there was still a negative dependency. Many more methods to overcome this class imbalance exist, such as different resampling techniques, which could be experimented with. It is expected that a gain in performance can thereby be reached.

Moreover, to limit the gap in performance when using a model trained on a disaster other than the given test disaster, one could explore techniques of unsupervised learning. An example of this kind of strategy could be pre-training the weights of the CNN through an autoencoder which uses the unlabeled satellite imagery of the test disaster as input [82].

Lastly, during this research, data of the xBD dataset was used, which has two limitations. Firstly, for this dataset solely disasters were selected with a high quality of satellite imagery. This is not always realistic, and thus the model should also be tested on disasters where the quality of imagery is less high. This will generate interesting insights in the sensitivity of the model to data quality. Secondly, the annotated damage data of the xBD dataset was gathered with the sole purpose to create the dataset, but this data was not used for an actual disaster response. To obtain the match between the classifications produced by the model and the annotated damage data used in current practices, the model should be tested on the imagery of a disaster for which damage data that was used for certain response action(s) is available. By doing this comparison with real used data, shortcomings of the model as well as of the current practices can be better understood.

7.3 ETHICAL CONSIDERATIONS

When designing a new tool, it is crucial to take into account its implications. While implications can never be fully calculated, a thought process on elements such as *data responsibility* and *equality* can be done. Regarding the data responsibility of the input data, in the current model satellite imagery is used as a source. This does not seem problematic regarding the privacy of the population in the impacted area, since the data is openly available and the spatial resolution is too low to identify individuals. In contrast to that, the data produced by the model, especially in combination with the building localization, holds larger implications. Knowing where buildings with which damage are located can also be used for other purposes than solely providing aid. An example in the past being that a government wanted to use the maps generated during the emergency response phase to plan an eviction campaign to get rid of illegal settlements. Thus, one should think if the generated information might lead to any negative implications and how these can be circumvented.

It is also important to understand which buildings are misclassified and ensure that this does not result in certain groups of people being hurt. This could, for example, happen if certain types of houses are more often misclassified by the model. The performance measure also plays a role here: one could choose to overclassify damage rather than underpredict it, to ensure that a larger percentage of damaged buildings are identified. Nevertheless, overestimating also has disadvantages, for example, it can lead to incorrect news or non-optimal allocation of funds.

It is crucial to anticipate users of the generated data and their possible purposes. It should then be decided together with the users, for which purposes what information should be communicated to whom and how. For example, when the generated numbers of damaged buildings are being used for marketing and raising funds, one wants to make sure that they reflect the truth and are being communicated clearly with the limitations of the method included. Moreover, it should be considered how the information provided by the tool is coupled with other essential information, such as data on the vulnerability and coping capacity of the impacted population, such that it can actually lead to more effective aid, the purpose of this research.

8 CONCLUSION

The goal of this research was to better understand the practical applicability of a CNN model for automated building damage assessment. The first step was to test if the designed CNN model is able to recognize damage from satellite imagery. The results show that this is the case and thus the model holds predictive value. By testing on a set of 13 disasters with four different damage types, it became clear however that the performance differs greatly per disaster. Testing on such a large set of disasters with such a variety in damage types has never been done before, and these differences show that it is very important to be aware that not every disaster is the same. An attempt was made to understand where these differences come from. One finding was that the class imbalance plays a role since a monotonically increasing dependency between the percentage of samples per class and the recall of that class was identified. Surprisingly, other disaster and image-specific parameters such as the geographical region, damage type, building footprint, and angles of the satellite were not found to have a significant impact on the used performance measures. From qualitative analysis, it was shown that the model has the most difficulty predicting ambiguous cases. For example blurry images, damage that is on the border of the definition of two classes, buildings where more surrounding area was needed to identify the damage, and images where the lightning gives difficulty. Moreover, it was shown that the model sometimes "outperforms" the human annotators by correctly classifying buildings that were wrongly labeled in the ground-truth data.

Furthermore, different granularities of classes were tested. A binary distinction between *destroyed* and *not destroyed* was made, which is the usual distinction in previous research, and a separation of four classes, *no damage, minor damage, major damage*, and *destroyed*, was made, since this more precise distinction seems to be needed for several purposes of the assessment. This distinction to four classes has not been researched before, with the argument that the *minor* and *major* classes are difficult to recognize from satellite imagery. This study shows that this statement is not valid for the given dataset, since those two intermediate classes were not performing significantly worse. Their performance, measured in the recall of that class, was not always high, but it is suspected that this is more a cause of the low percentage of buildings belonging to those classes than the specific damage features. In addition, it was shown that even when solely the binary distinction is wanted, it is still better to train the model on the multiclass labels. After training they can then be grouped to binary labels, resulting in a better separability between the two classes compared to when the model is trained on binary labels.

This research also dived into the question of the transferability of a learned model to a new disaster. For this experiment, the test disaster was held constant while different sets of training data were created that did not include the test disaster. In this research, two different disasters were taken as test disasters. The Joplin tornado, which struck in the USA, caused wind damage and gained high values for the used performance measures when trained on itself, and the Nepal flooding, which struck in another part of the world with a different damage type, namely flooding compared to wind, and did not gain good values of the performance measures when trained on itself. Differences in transferability were striking, where the best set of train disasters for Joplin reached 93% of the original macro F1 measure while this was only 53% for the Nepal flooding. In both cases, the best performance was reached when only training on disasters with the same damage type. Based on the

results, it is suggested that it is the best to include all disasters available with the same damage type as the test disaster, since this gives the most security on consistent, though not always the best results. Reaching 93% of the performance for the Joplin tornado is a surprisingly good result and would imply that this is a setting that is applicable in practice. However, due to the disappointing values of the performance measures on the Nepal flooding, the cause of differences in transferability and the optimal composition of training samples has to be studied further. Nevertheless, the promising results of the Joplin tornado show that automated damage assessment has a lot of potential, also in the context of realistic data requirements.

To further improve the performance of the model several modifications were done. It was shown that resampling the training data such that all classes are balanced, and adding to the loss a weight that is related to the inverse class frequency, both improve the ability to recognize the minority classes. This is a positive result since without these modifications the model experienced difficulties recognizing the minority classes, while for real applications these are often the most important to recognize well. Based on these results, it is advised to always implement one of these methods to handle imbalance and to even look into other methods to diminish those effects more. Besides the modifications for imbalance, interesting results were gained when simplifying the model. Removing the data augmentation gave worse results, but there were only little differences. The same applies to solely using post imagery instead of a combination of pre and post imagery, where the observed small drop in performance is consistent with the previous research. Therefore, it is suggested to continue using pre and post imagery if available, but also look into the potential the model might hold if solely high-quality post imagery is at hand.

For the tool to be ready for use in practice, user needs should be more thoroughly researched and it should be better understood on which disasters a good result in terms of the performance measures can be reached and on which not. Nevertheless, this study showed that when those factors are better understood, there is a lot of potential for the automated damage assessment tool to provide added value when applied in practice.

BIBLIOGRAPHY

[1] A. Lacoste, A. Luccioni, V. Schmidt, and T. Dandres, "Quantifying the carbon emissions of machine learning," *arXiv preprint arXiv:1910.09700*, 2019.

[2] D. Guha-Sapir, P. Hoyois, P. Wallemacq, and R. Below, "Annual disaster statistical review 2016: The numbers and trends," Centre for Research on the Epidemiology of Disasters, 2016.

[3] H. Huppert and S. Sparks, "Extreme natural hazards: population growth, globalization and environmental change," *Philosophical Transactions of the Royal Society A*, vol. 364, no. 1845, 2006.

[4] G. Shen, L. Zhou, Y. Wu, and Z. Cai, "A global expected risk analysis of fatalities, injuries, and damages by natural disasters," *Sustainability*, vol. 10, no. 7, 2018.

[5] World Bank and United Nations, *Natural Hazards, Unnatural Disasters: The Economics of Effective Prevention*. Washington D.C.: The World Bank, 2010.

[6] P. J. Klotzbach, S. G. Bowen, R. Pielke Jr, and M. Bell, "Continental US hurricane landfall frequency and associated damage: Observations and future risks," *Bulletin of the American Meteorological Society*, vol. 99, no. 7, 2018.

[7] D. P. Coppola, *Introduction to international disaster management*, 3rd ed. Butterworth-Heinemann, 2015.

[8] D. Guha-Sapir and P. Hoyois, "Estimating populations affected by disasters: A review of methodological issues and research gaps," Centre for Research on the Epidemiology of Disasters, Institute of Health and Society, University Catholique de Louvain, 2015.

[9] 510, an Initiative of the Netherlands Red Cross. (2016) 510 mission & vision. [Online]. Available: https://www.510.global/510-mission-vision/

[10] B. Bronmans, "Building detection on aerial imagery using convolutional neural networks," Master's thesis, Vrije Universiteit Amsterdam, 2019.

[11] D. Kersbergen, "Automated building damage classification using remotely sensed data: Case study: Hurricane damage on st. maarten," Master's thesis, Delft University of Technology, 2018.

[12] O. Courtin and D. J. Hofmann, *neat-EO Efficient AI4EO OpenSource framework*, DataPink, 2020. [Online]. Available: http://neat-EO.pink

[13] Centre for Research on the Epidemiology of Disasters. (2009) EM-DAT glossary. [Online]. Available: https://www.emdat.be/Glossary

[14] International Federation of Red Cross and Red Crescent Societies. (2019) About disasters. [Online]. Available: https://www.ifrc.org/en/what-we-do/disaster-management/about-disasters/

[15] D. Alexander, *Principles of Emergency Planning and Management*. Oxford University Press, 2002.

[16] European Commission and the United National Development Group and World Bank. (2013) Post-disaster needs assessments guidelines volume B housing.

[17] S. Loos, K. Barns, G. Bhattacharjee, R. Soden, B. Herfort, M. Eckle, C. Giovando, B. Girardot, G. Deierlein, A. Kiremidjian, J. Baker, and D. Lallemant, "The development and uses of crowdsourced building damage information based on remote-sensing," Blume Center, Tech. Rep., 2018.

[18] M. Meaney and M. Flores, "Habitat's for humanity disaster risk reduction and response; pathways to permanence," Habitat for Humanity, Tech. Rep.

[19] H. R. Ranjbar, A. A. Ardalan, H. Dehghani, and M. R. Saradjian, "A proposed spatial index to prioritize damaged buildings for allocating USAR operations," *Geocarto International*, vol. 33, no. 8, 2018.

[20] I. Petiteville, S. Ward, G. Dyke, M. Steventon, and J. Harry, "Satellite earth observations in support of disaster risk reduction," in *The CEOS Earth Observation Handbook, Special 2015 Edition for the 3rd UN World Conference on Disaster Risk Reduction*, 2015.

[21] T. Jeggle and M. Boggero, "Post-disaster needs assessment (PDNA): Lessons from a decade of experience," Washington, D.C.: World Bank, 2018.

[22] M. Van den Homberg and I. Susha, "Characterizing data ecosystems to support official statistics with open mapping data for reporting on sustainable development goals," *ISPRS International Journal of Geo-Information*, vol. 7, no. 12, 2018.

[23] S. Voigt, F. Giulio-Tonolo, J. Lyons, J. Kučera, B. Jones, T. Schneiderhan, G. Platzeck, K. Kaku, M. K. Hazarika, L. Czaran *et al.*, "Global trends in satellite-based emergency mapping," *Science*, vol. 353, no. 6296, 2016.

[24] P. Boccardo and F. Giulio Tonolo, "Haiti earthquake damage assessment: Review of the remote sensing role," *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 39, no. 9, 2012.

[25] C. Westrope, R. Banick, and M. Levine, "Groundtruthing openstreetmap building damage assessment," *Procedia engineering*, vol. 78, 2014.

[26] Post Disaster Needs Assessment Team, Government of Republic of Haiti. (2010) Haiti earthquake PDNA: assessment of damage, losses, general and sectoral needs.

[27] L. Dell'Oro. (2011) Geospatial approaches to damage assessment: The example of haiti earthquake. UNITAR-UNOSAT. [Online]. Available: https://www.preventionweb.net/files/globalplatform/entry_presentation~1lucawrcitinnovationsgeospatialapproachestodamageassessmenttheexampleofhaitildov1.pdf

[28] N. Kerle, F. Nex, M. Gerke, D. Duarte, and A. Vetrivel, "UAV-based structural damage mapping: A review," *ISPRS international journal of geo-information*, vol. 9, 2019.

[29] T. Brown, *Change by design: how design thinking transforms organizations and inspires innovation*. New York: Harper Business, 2009.

[30] G. Grünthal, R. Musson, J. Schwarz, and M. Stucchi, "EMS-98 (european macroseismic scale)," European Seismological Commision, Tech. Rep., 1998.

[31] S. Cotrufo, C. Sandu, F. Giulio Tonolo, and P. Boccardo, "Building damage assessment scale tailored to remote sensing vertical imagery," *European Journal of Remote Sensing*, vol. 51, no. 1, 2018.

[32] F. Dell'Acqua and P. Gamba, "Remote sensing and earthquake damage assessment: Experiences, limits, and perspectives," *Proc. IEEE*, vol. 100, no. 10, 2012.

[33] R. Gupta, B. Goodman, N. Patel, R. Hosfelt, S. Sajeev, E. Heim, J. Doshi, K. Lucas, H. Choset, and M. Gaston, "Creating xBD: A dataset for assessing building damage from satellite imagery," in *Proc. the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2019.

[34] J. B. Campbell and R. H. Wynne, *Introduction to remote sensing*. Guilford Press, 2011.

[35] L. Ma, Y. Liu, X. Zhang, Y. Ye, G. Yin, and B. A. Johnson, "Deep learning in remote sensing applications: A meta-analysis and review," *ISPRS journal of photogrammetry and remote sensing*, vol. 152, 2019.

[36] F. Vescovi and A. Minchella, "CSCDA - coordinated data-access system (CDS)," European Space Agency, Tech. Rep. 3, 2017.

[37] N. Gorelick, M. Hancher, M. Dixon, S. Ilyushchenko, D. Thau, and R. Moore, "Google earth engine: Planetary-scale geospatial analysis for everyone," *Remote Sensing of Environment*, vol. 202, 2017.

[38] DigitalGlobe. (2020) Tools & resources. [Online]. Available: https://www.digitalglobe.com/resources

[39] A. Sharma, X. Liu, X. Yang, and D. Shi, "A patch-based convolutional neural network for remote sensing image classification," *Neural Networks*, vol. 95, 2017.

[40] N. Jean, S. Wang, A. Samar, G. Azzari, D. Lobell, and S. Ermon, "Tile2vec: Unsupervised representation learning for spatially distributed data," in *Proc. AAAI Conference on Artificial Intelligence*, vol. 33, 2019.

[41] F. Nex, D. Duarte, F. G. Tonolo, and N. Kerle, "Structural building damage detection with deep learning: Assessment of a state-of-the-art CNN in operational conditions," *Remote sensing*, vol. 11, no. 23, 2019.

[42] J. Z. Xu, W. Lu, Z. Li, P. Khaitan, and V. Zaytseva, "Building damage detection in satellite imagery using convolutional neural networks," *arXiv preprint arXiv:1910.06444*, 2019.

[43] A. Weinert, "Video and imagery dataset to drive public safety capabilities," in *Public Safety Broadband Stakeholder Meeting*, Chicago, IL, USA, Jul. 2019. [Online]. Available: https://cdnapisec.kaltura.com/index.php/extwidget/preview/partner_id/684682/uiconf_id/31013851/entry_id/1_tux4px6k/embed/dynamic

[44] S. Yun, K. Hudnut, S. Owen, F. Webb, M. Simons, P. Sacco, E. Gurrola, G. Manipon, C. Liang, E. Fielding *et al.*, "Rapid damage mapping for the 2015 Mw 7.8 Gorkha earthquake using synthetic aperture radar data from COSMO-SkyMed and ALOS-2 satellites," *Seismological Research Letters*, vol. 86, no. 6, 2015.

[45] C. Lu, C. Ni, C. Chang, J. Yen, and R. Chuang, "Coherence difference analysis of Sentinel-1 SAR interferogram to identify earthquake-induced disasters in urban areas," *Remote Sensing*, vol. 10, no. 8, 2018.

[46] H. Gokon, J. Post, E. Stein, S. Martinis, A. Twele, M. Mück, C. Geiß, S. Koshimura, and M. Matsuoka, "A method for detecting buildings destroyed by the 2011 Tohoku earthquake and tsunami using multitemporal TerraSAR-X data," *IEEE Geoscience and Remote Sensing Letters*, vol. 12, no. 6, 2015.

[47] M. Ji, L. Liu, R. Du, and M. F. Buchroithner, "A comparative study of texture and convolutional neural network features for detecting collapsed buildings after earthquakes using pre-and post-event satellite imagery," *Remote Sensing*, vol. 11, no. 10, 2019.

[48] A. J. Cooner, Y. Shao, and J. B. Campbell, "Detection of urban damage using remote sensing and machine learning algorithms: Revisiting the 2010 haiti earthquake," *Remote Sensing*, vol. 8, no. 10, 2016.

[49] A. Fujita, K. Sakurada, T. Imaizumi, R. Ito, S. Hikosaka, and R. Nakamura, "Damage detection from aerial images via convolutional neural networks," in *Proc. Fifteenth IAPR International Conference on Machine Vision Applications (MVA)*, 2017.

[50] Q. D. Cao and Y. Choe. (2019) Building damage annotation on post-hurricane satellite imagery based on convolutional neural networks. ArXiv:1807.01688v3.

[51] M. Ji, L. Liu, R. Zhang, and M. F Buchroithner, "Discrimination of earthquake-induced building destruction from space using a pretrained CNN model," *Applied Sciences*, vol. 10, no. 2, 2020.

[52] A. C. Jung, "Machine learning: Basic principles." 2018, arXiv:1805.05052.

[53] F. van Veen and S. Leijnen. (2019) The neural network zoo. [Online]. Available: https://www.asimovinstitute.org/neural-network-zoo

[54] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012.

[55] B. T. Polyak, "Some methods of speeding up the convergence of iteration methods," *USSR Computational Mathematics and Mathematical Physics*, vol. 4, no. 5, 1964.

[56] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[57] T. Tieleman and G. Hinton, "Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude," *COURSERA: Neural networks for machine learning*, vol. 4, no. 2, 2012.

[58] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. 3rd International Conference on Learning Representations ICLR*, 2015.

[59] T. Hastie, R. Tibshirani, and J. Friedman, *The elements of statistical learning: data mining, inference, and prediction*. Springer Science+Business Media, 2009.

[60] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *The journal of machine learning research*, vol. 15, no. 1, 2014.

[61] J. Gu, Z. Wang, J. Kuen, L. Ma, A. Shahroudy, B. Shuai, T. Liu, X. Wang, G. Wang, J. Cai, and T. Chen, "Recent advances in convolutional neural networks," *Pattern Recognition*, vol. 77, 2018.

[62] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. 32nd International Conference on Machine Learning*, 2015.

[63] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, 2015.

[64] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *ICLR*, 2015.

[65] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE conference on computer vision and pattern recognition*, 2015.

[66] S. Tsang. (2018) Review: Inception-v3 - 1st runner up (image classification) in ILSVRC 2015. [Online]. Available: https://medium.com/@sh.tsang/review-inception-v3-1st-runner-up-image-classification-in-ilsvrc-2015-17915421f77c

[67] S. Ruder. (2017) Transfer learning - machine learning's next frontier. [Online]. Available: https://ruder.io/transfer-learning/

[68] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Transactions on knowledge and data engineering*, vol. 22, no. 10, 2009.

[69] M. T. Rosenstein, Z. Marx, L. P. Kaelbling, and T. G. Dietterich, "To transfer or not to transfer," in *NIPS 2005 workshop on transfer learning*, vol. 898, 2005.

[70] P. Agrawal, R. Girshick, and J. Malik, "Analyzing the performance of multilayer neural networks for object recognition," in *Proc. ECCV*, 2014.

[71] U. Kamath, J. Liu, and J. Whitaker, *Transfer Learning: Domain Adaptation*. Cham: Springer International Publishing, 2019, pp. 495–535.

[72] M. Buda, A. Maki, and M. A. Mazurowski, "A systematic study of the class imbalance problem in convolutional neural networks," *Neural Networks*, vol. 106, 2018.

[73] B. Krawczyk, "Learning from imbalanced data: open challenges and future directions," *Progress in Artificial Intelligence*, vol. 5, no. 4, 2016.

[74] J. M. Johnson and T. M. Khoshgoftaar, "Survey on deep learning with class imbalance," *Journal of Big Data*, vol. 6, no. 1, 2019.

[75] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: synthetic minority over-sampling technique," *Journal of artificial intelligence research*, vol. 16, 2002.

[76] M. Kukar, I. Kononenko *et al.*, "Cost-sensitive learning with neural networks." in *ECAI*, vol. 98, 1998.

[77] S. H. Khan, M. Hayat, M. Bennamoun, F. A. Sohel, and R. Togneri, "Cost-sensitive learning of deep feature representations from imbalanced data," *IEEE transactions on neural networks and learning systems*, vol. 29, no. 8, 2017.

[78] A. P. Bradley, "The use of the area under the roc curve in the evaluation of machine learning algorithms," *Pattern recognition*, vol. 30, no. 7, 1997.

[79] A. Vetrivel, M. Gerke, N. Kerle, F. Nex, and G. Vosselman, "Disaster damage detection through synergistic use of deep learning and 3D point cloud features derived from very high resolution oblique aerial images, and multiple-kernel-learning," *ISPRS journal of photogrammetry and remote sensing*, vol. 140, 2018.

[80] A. V. Etten, D. Lindenbaum, and T. M. Bacastow, "Spacenet: A remote sensing dataset and challenge series," 2018, arXiv:1807.01232.

[81] T. Fushiki, "Estimation of prediction error by using K-fold cross-validation," *Statistics and Computing*, vol. 21, no. 2, 2011.

[82] Y. Li, W. Hu, H. Dong, and X. Zhang, "Building damage detection from post-event aerial imagery using single shot multibox detector," *Applied Sciences*, vol. 9, no. 6, 2019.